معاونت علمی و فناوری ریاست جمهوری

ستاد توسعه زیست‌فناوری

کاربرد نرم‌افزار R در تجزیه تنوع ژنتیکی و ژنتیک جمعیت

تألیف:

دکتر مهدی زهراوی


بسم الله الرحمن الرحيم

فهرست مطالب

عنوان **صفحه**

عنوان	صفحه

فصل چهارم- آمار توصیفی داده‌های ژنتیکی و آزمون‌های پایه

عنوان	صفحه

فصل پنجم- فواصل ژنتیکی

فصل ششم- آماره‌های افتراق جمعیت

فصل هفتم- تجزیه چندمتغیره داده‌های ژنتیکی

عنوان	صفحه

فصل هشتم- بررسی سطوح ساختاری جمیعت

فصل نهم- شناسایی الگوهای ژنتیکی- جغرافیایی

فصل دهم- داده‌های توالی

عنوان	صفحه

پیش‌گفتار مؤلف

تنوع ژنتیکی بیانگر تفاوت‌های ژنتیکی در بین افراد هر گونه زیستی است و نقش مهمی در ماندگاری و سازگاری آنها ایفاء می‌کند. از این رو حفظ تغییرپذیری ژنتیکی به منزله شرط اصلی بقاء یا انقراض هر گونه‌ای در برابر نوسانات محیطی به شمار می‌آید. کاهش تنوع ژنتیکی، برخلاف رخدادهای نابودی و انقراض، پدیده‌ی آشکاری نیست و آگاهی از آن مستلزم بکارگیری روش‌های دقیقِ مشاهده و ارزیابی می‌باشد. این حقیقت از سویی اهمیت پایش تغییرات میزان تنوع ژنتیکی در گونه‌ها را نشان می‌دهد و از سوی دیگر لزوم بهره‌برداری از روش‌های کارآمد در ارزیابی این تنوع و بررسی عوامل اثرگذار برآن را آشکار می‌نماید. این روش‌ها در قالب زمینه‌ای از دانش "ژنتیک جمعیت" توسعه یافته‌اند. بنابراین حفاظت از تنوع ژنتیکی گونه‌ها مستلزم درک و بکارگیری مدل‌های ارزیابی تغییرات ژنی و آللی در جمعیت‌ها در زمان و مکان می‌باشد. بدین ترتیب، دانش "ژنتیک جمعیت" ماهیتی کمّی داشته و محاسبات، به عنوان جزء اساسی آن به شمار می‌روند. با گسترش مدل‌های ریاضی پیچیده، این خطر ایجاد شده است که چالش‌های محاسباتی و برنامه‌نویسی، دامنه‌ی گسترش، عمق و یا کیفیت تحقیقات مرتبط در این زمینه را با محدودیت مواجه سازد. از سویی دیگر پیشرفت فناوری، حجم داده‌های ایجاد شده در هر آزمایش را به میزان چشمگیری افزایش داده و توسعه نرم‌افزارهایی با قابلیت پردازش کارآمد را اجتناب ناپذیر ساخته است. در این رابطه R به عنوان یکی از قدرتمندترین نرم‌افزارهای محاسباتی در برآوردن این نیازها بسیار نوید بخش ظاهر شده است. ویژگی نرم‌افزار R، امکان نوشتن کدهای کارآمد، اجرای تکنیک‌های محاسباتی مختلف و افزایش سرعت چشمگیر سرعت محاسبات را میسر ساخته است. از آنجا که تجزیه داده‌های ژنتیکی می‌تواند از جنبه‌های مختلفی مدنظر قرار گرفته و به روش‌های متنوعی انجام گیرد، پکیج‌های متنوع تجزیه ژنتیکی با پرداختن به زمینه‌های اختصاصی، در ذیل نرم‌افزار R توسعه یافته‌اند. با این وجود، توابعی با کارکرد مشابه نیز در پکیج‌های متفاوت مشاهده می‌شود که خروجی آنها در مجموع، مکمل یکدیگر خواهد بود. کتاب حاضر با هدف تجمیع توابع مفید در قالب یک مجموعه و تشریح کارکرد و مقایسه کارایی آنها تدوین گردید. باید توجه داشت که هر یک از پکیج‌های مذکور، خود مشتمل بر بسیاری از توابع می‌باشند و

لذا شرح و تفصیل تمام آنها در این کتاب مدنظر نبوده و خوانندگان محترم برای جزئیات بیشتر بایستی به راهنمای مربوطه موجود در اینترنت مراجعه نمایند. مطالب این کتاب برای انجام تجزیه‌های ژنتیکی در ارگانیسم‌های مختلف اعم از جمعیت‌های انسانی، گیاهی، جانوری و ریزسازواره‌ها قابل استفاده می‌باشد و به همین دلیل سعی گردیده در مثال‌هایی که برای تشریح توابع ذکر شده، از انواع متنوعی از داده‌ها استفاده شود.

مؤلف وظیفه خود می‌داند از ستاد توسعه زیست‌فناوری و گروه پژوهش، توسعه و زیرساخت فناوری جهت حمایت از چاپ این مجموعه، سپاسگزاری نماید.

مهدی زهراوی

زمستان ۱۳۹۶

فصل اول-مقدمه

ژنتیک جمعیت و اهمیت آن

ژنتیک جمعیت شاخه‌ای از علم زیست‌شناسی است و عبارت از دانش مطالعه تنوع ژنتیکی در درون جمعیت‌ها می‌باشد. این حوزه از علم به منشاء، میزان و توزیع تغییرات ژنتیکی در جمعیت‌های موجودات زنده که از عوامل گوناگونی مانند گزینش طبیعی ناشی شده‌اند، می‌پردازد و سرنوشت این تغییرات را در زمان و مکان مورد مطالعه قرار می‌دهد. بسیاری از ژن‌ها در داخل یک جمعیت پلی‌مورفیک می‌باشند، یعنی به فرم‌های مختلف (یا آلل‌ها) حضور دارند. با استفاده از مدل‌های ریاضی می‌توان حضور آلل‌ها یا ترکیبی از آن‌ها را در جمعیت پیش‌بینی نمود. این پیش‌بینی برپایه درک اساس مولکولی ژنتیک، قوانین توارث مندل و نظریه نوین تکامل، استوار می‌باشد. ژنتیک جمعیت بر جمعیت‌ها یا گونه‌ها و نه بر افراد، متمرکز است (Ewens, 2012 ؛Gillespie, 2010 ؛Christiansen, 1999).

تغییرات ژن‌ها در زمان و مکان اساس تغییرات تکاملی را تشکیل می‌دهد. بدین ترتیب ژنتیک جمعیت ارتباط تنگاتنگی با مطالعه تکامل و گزینش طبیعی دارد. گزینش طبیعی یکی از مهمترین عواملی است که ترکیب ژنتیکی جمعیت را تحت تأثیر قرار می‌دهد. گزینش طبیعی وقتی رخ می‌دهد که برخی از واریانت‌های درون جمعیت، سازگاری بیشتری به محیط داشته باشد. با فرض اینکه تفاوت در شایستگی حداقل تا حدود زیادی از تفاوت‌های ژنتیکی نشأت می‌گیرد، آنگاه قابل تصور خواهد بود که ترکیب ژنتیکی جمعیت طی زمان تغییر کند. بنابراین متخصصان ژنتیک جمعیت با مطالعه مدل‌های تغییر فراوانی ژنی امیدوارند که به درک بهتری از فرایندهای تکاملی نائل آیند. لذا ژنتیک جمعیت در قلب زیست‌شناسی تکاملی قرار دارد و می‌توان آن را به عنوان علم سازوکارهای مسؤل تکامل درون‌گونه‌ای (تکامل میکرو[1])

[1] Microevolution

قلمداد کرد. بسیاری از این سازکارها تأثیر زیادی روی نشأت گرفتن گونه جدید و همچنین بر روی تکامل فراتر از سطح گونه (تکامل ماکرو[۲]) دارند (Ayala, 1982؛ Gillespie, 1994؛ Höglund, 2009؛ Mettler and Gregg 1969؛ Templeton, 2006).

آنچه ژنتیک جمعیت را در بین سایر علوم طبیعی منحصر به فرد می‌سازد، رویکرد آن در قبال سؤالات پیرامون جهان زیستی است. ژنتیک جمعیت به مثابه محاوره‌ای مابین پیش‌بینی‌های مبتنی بر اصول توارثی مندلی و نتایج حاصل از ارزیابی‌های تجربیِ فراوانی‌های ژنوتیپی و آللی، که اساس فرضیات آزمون را تشکیل می‌دهند، می‌باشد. پیش‌بینی‌های ایده‌آل برآمده از اصول کلی، اساس فرضیاتی را تشکیل می‌دهند که می‌توانند بعداً آزمون شوند. در عین حال الگوهای تجربی مشاهده شده در بین یا درون جمعیت‌ها، با مقایسه فرایندهایی که ممکن است چنین الگوهایی را ایجاد کرده باشند، قابل توضیح هستند (Hamilton, 2011).

ژنتیک جمعیت در دهه‌های ۱۹۲۰ و ۱۹۳۰ در پی تلاش‌های سه دانشمند مشهور، فیشر[۳]، هالدین[۴] و سول رایت[۵]، پا به عرصه ظهور گذاشت. چارچوب نظری این رشته علمی طی سال‌های ۱۹۲۰ الی ۱۹۸۰ توسعه یافت و در این بازه زمانی، پیش‌بینی‌های زیادی پیرامون فرایندهای تکاملی پایه صورت پذیرفت. اما اکثر این پیش‌بینی‌ها قابل آزمون نبودند و یا اینکه به دلیل عدم کفایت داده‌های ژنتیکی، آزمون‌های انجام شده از قدرتمندی کافی برخوردار نبودند. بدین ترتیب ژنتیک جمعیت در قرن بیستم سرشار از فرضیات بوده، ولی در عین حال به کمبود داده‌های آزمایشی دچار بوده است. اما هم اکنون دانش ژنتیک جمعیت با محدودیت داده مواجه نیست. پیشرفت اصلی در عرصه ژنتیک جمعیت در طی دهه‌های گذشته ناشی از دسترسی به مقادیر عظیمی از داده‌های ژنتیکی می‌باشد که حاصل توانایی در توالی‌یابی کل ژنوم در موجودات زنده بوده است. در حال حاضر با افزایش توانمندی در جمع‌آوری سریع حجم انبوهی از اطلاعات ژنتیکی تقریباً در هر موجود زنده‌ای، و تسهیم رایگان و گسترده آن‌ها، ژنتیک جمعیت از نظر داده غنی شده است. با بلوغ ژنتیک مولکولی و گام نهادن آن به عرصه ژنومیکس، ژنتیک جمعیت از جایگاه صرفِ استفاده کننده از ماحصل تکنیک‌های نوین ژنتیک مولکولی تغییر نقش داده و به عنوان دانشی تحول یافته، روش‌های تحلیلی پایه را برای بسیاری از جنبه‌های ژنومیکس فراهم آورده است (Crow and Kimura, 1970؛ Doolittle, 2012؛ Hamilton, 2011؛ Hartl, 1988؛ Hedrick, 2011).

ژنتیک جمعیت از هر دو روش استدلال استقرایی و قیاسی برای درک عمل فرایندهای زیستی در جمعیت‌های واقعی و همچنین برای شناسایی فرایندهای عمومی که سبب رویداد پدیده‌های ژنتیکی

[2] Macroevolution
[3] R.A.Fisher
[4] J.B.S. Haldane
[5] Swall Wright

می‌شوند، استفاده می‌کند. روش استقرایی در ژنتیک جمعیت شامل جمع‌آوری معیارهای تنوع ژنتیکی (برآورد پارامترها) از جمعیت‌های گوناگون به منظور دستیابی به شواهد یا دلایل است. از این شواهد برای شناسایی فرایندهایی استفاده می‌شود که الگوهای مشاهده شده را ایجاد کرده‌اند. بکارگیری استدلال استقرایی نیازمند آشنایی عمیق با انواع متنوع داده‌ها در ژنتیک جمعیت، از جمله داده‌های مربوط به توالی‌های DNA می‌باشد. این امر همچنین مستلزم مدنظر قرار دادن نتایج مطالعاتی است که الگوهای مشاهده شده در تنوع ژنتیکی را گزارش می‌نمایند. با تجمع تدریجی این مشاهدات و اطلاعات تجربی می‌توان نتایج کلی‌تری را پیرامون کمیّت و کیفیت تنوع ژنتیکی در جمعیت‌ها، استنتاج نمود (Hamilton, 2011).

مطالعات ژنتیک جمعیت همچنین می‌تواند با رویکرد استدلال قیاسی صورت گیرد. در این حالت، عمل فرایندهای تکاملی مانند رانده شدن ژنتیکی و گزینش طبیعی بصورت پارامترهایی در قالب فرمول‌های ریاضی نشان داده می‌شوند. این فرمول‌های ریاضی، مدل‌های ژنتیکی را تشکیل می‌دهند و با استفاده از آن‌ها می‌توان میزان تنوع ژنتیکی و الگوی آن در زمان و مکان را پیش‌بینی نمود. چنین مدل‌هایی به عنوان مثال امکان پیش‌بینی میزان تغییر فراوانی‌های آللی و حاصل برآیند فرایندهای گوناگون که بطور همزمان عمل می‌کنند را، فراهم می‌سازند. این پیش‌بینی‌ها از عمومیت برخوردار هستند و در هرگونه‌ای قابل استفاده می‌باشند. اما در عین حال ممکن است برای جمعیت‌های خاص به کار نیایند، زیرا اصول و فرضیات کلی که برای انجام چنین پیش‌بینی‌هایی مورد استفاده قرار گرفته‌اند از اختصاصیت کافی برای تطابق با جمعیت‌های واقعی، برخوردار نمی‌باشند (Corander *et al.*, 2003؛ Hamilton, 2011؛ Pritchard *et al.*, 2000).

یکی از موارد مهم کاربرد ژنتیک جمعیت، مطالعات مربوط به حفاظت از گونه‌ها می‌باشد. امروزه شاهد گسترش توجه جهانی به بحران انقراض گونه‌ها، تخریب و نابودی زیستگاه‌های طبیعی و تغییرات اقلیمی می‌باشیم. در این راستا ژنتیک جمعیت ابزارهای زیادی برای مطالعه زیست‌شناسی حفاظت در اختیار می‌گذارد. این عوامل سبب وسیع شدن دامنه مخاطبان دانش ژنتیک جمعیت شده است. بدین ترتیب دانش ژنتیک حفاظت[۶] برمبنای پیش‌بینی‌های حاصل از مدل‌های ژنتیکی جمعیت استوار شده است. فرض اساسی ژنتیک حفاظت آن است که جمعیت‌های کوچک در معرض خطر می‌باشند. این فرض مبتنی بر این پیش‌بینیِ ژنتیکیِ جمعیت است که رانده شدنِ ژنتیکیِ تصادفی[۷] و درون‌زادآوری[۸]، تنوع آللی و ژنوتیپی جمعیت‌ها را به عنوان تابعی از اندازه‌ی جمعیت‌های مذکور تحت تأثیر قرار می‌دهد. نظریه ژنتیک حفاظت در سال‌های اخیر در دو زمینه تحول یافته است. زمینه اول عبارت از تعداد رو به افزایش مدل‌هایی است که

[6] Conservation Genetics
[7] Random genetic drift
[8] Inbreeding

تاریخچه زیست جمعیت‌ها را با وضعیت جغرافیایی محل استقرار آنها تلفیق می‌کنند و بدین ترتیب پیش‌بینی دقیق‌تری از دینامیک تنوع ژنتیکی در زمان و مکان فراهم می‌آورند. نتایج مطالعه براساس این مدل‌ها نشان داده است که با تلفیق واقعیت‌های زیستی می‌توانیم درک و دقت پیش‌بینی خود را از سرنوشت جمعیت‌ها و گونه‌ها بهبود بخشیم. تحول دوم در زمینه ژنتیک حفاظت، توسعه روش‌های پیشرفته در تجزیه داده‌های ژنتیکی است. این دستاورد شدیداً متأثر از ظهور نشانگرهای مولکولی جدید با قدرت تفکیک‌پذیری بالاست که هریک دارای ویژگی‌های ژنتیکی خاصی می‌باشند (Allendorf and Luikart, 2007؛ Allendorf and Luikart, 2009؛ Amato *et al.*, 2009؛ Conner and Hartl, 2004؛ Falk and Holsinger, 1991؛ Frankham *et al.*, 2002؛ Ouborg, 2009؛ Schonewald-Cox *et al.*,1983؛ Schonewald *et al.*, 2003).

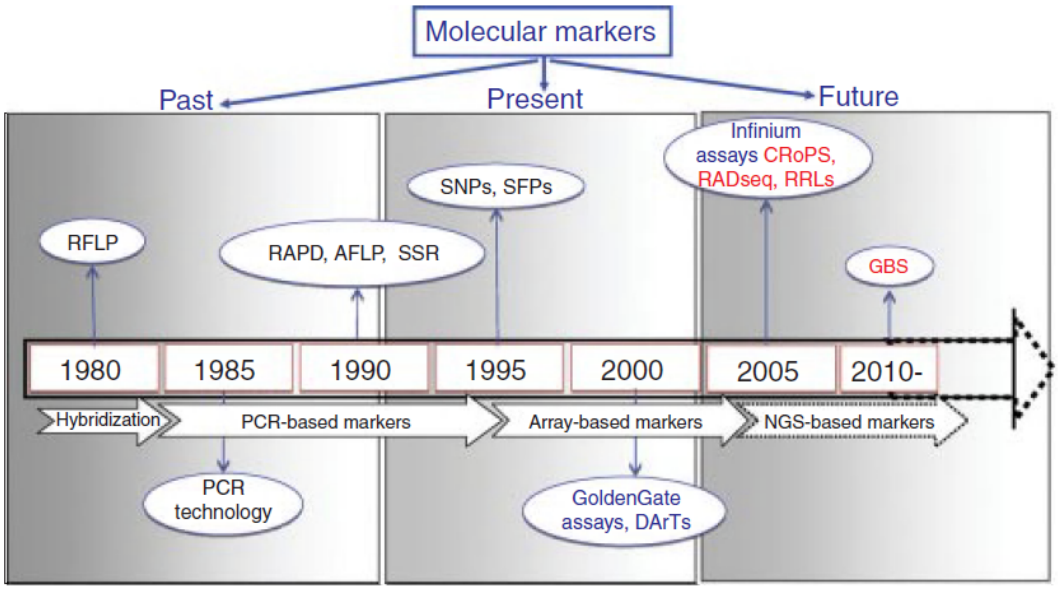
نشانگرهای مولکولی و زمینه‌های کاربرد آن‌ها

نشانگرهای مولکولی از دامنه کاربرد وسیعی در سیستم‌های زیستی برخوردار بوده و ابزار کلیدی در شناسایی ژرم‌پلاسم، پی بردن به روابط خویشاوندی و انجام اصلاح ژنتیکی می‌باشند. تعیین روابط خویشاوندی، پیش‌نیازِ تجزیه‌های ژنتیکی تکامل و حفاظت می‌باشد. انگشت‌نگاری، تهیه شناسنامه و شناسایی ژنتیکی ژرم‌پلاسم در اصلاح نژاد، تولید و فرآوری محصول، حقوق مالکیت فکری، و زمینه‌های پزشکی قانونی از اهمیت زیادی برخوردار است. نشانگرهای مولکولی را می‌توان به طور مستقیم در برنامه‌های اصلاح نژاد مورد استفاده قرار داد. نشانگرهای مولکولی می‌توانند کارایی و سرعت به‌نژادی را به طور توأم افزایش دهند. گزینش مبتنی بر نشانگر در مقایسه با روش‌های سنتی از مزایای زیادی از جمله سهولت بیشتر (هنگامی که ثبت مشخصات ظاهری دشوار است)، سرعت بالاتر (زمانی که تعیین خصوصیات فنوتیپی زمان‌بر است مثلاً مواقعی که نیاز به آزمون نتاج باشد)، ارزان‌تر (به شرط پایین‌تر بودن هزینه ژنوتیپ‌یابی نسبت به تعیین فنوتیپ) و کارآمدی بیشتر (به شرطی که توارث‌پذیری نشانگر صد در صد بوده و خطای ژنوتیپ‌یابی وجود نداشته باشد) می‌باشد. نشانگرهای مولکولی علاوه بر موارد کاربردی، در حوزه دانش بنیادین و خصوصاً فهم معماری ژنتیکی صفات پیچیده بسیار مفید می‌باشند.

با استفاده از نشانگرهای مولکولی می‌توان به سؤالات اساسی پیرامون جمعیت‌های طبیعی موجودات زنده پاسخ داد. الگوی توزیع تنوع ژنتیکی در این جمعیت‌ها همیشه مورد سؤال می‌باشد، بطور مثال اینکه جمعیت‌ها از تنوع ژنتیکی بالایی برخوردارند یا خیر، جمعیت‌ها به طور یکنواخت توزیع شده‌اند و یا متشکل از گروه‌های مجزا می‌باشند و آیا تفاوت ژنتیکی بین آنها وجود دارد؟ آیا جریان ژنی بین جمعیت‌ها وجود

دارد و الی آخر. اطلاعات به دست آمده از میزان تنوع ژنتیکی و الگوی توزیع آن، در شناسایی مراکز تنوع و اتخاذ راهبردهای کارآمد در زمینه‌ی نمونه‌برداری و جمع‌آوری ذخایر ژنتیکی بسیار مفید می‌باشد. پایش و مدیریت منابع ژنتیکی در زیستگاه‌های طبیعی نیازمند تعریف مقیاس‌های جغرافیایی مناسب است که براساس مطالعات تنوع ژنتیکی، قابل دستیابی می‌باشد. پیش نیاز حفاظت از ذخایر ژنتیکی، شناسایی جمعیت‌هایی است که از تاریخ تکاملی متمایزی برخوردار می‌باشند. همچنین در مدیریت جمعیت‌ها عواملی مانند نظام آمیزش، پسروی ناشی از درون‌زادآوری، اندازه جمعیت مؤثر و تقسیمات فرعی در جمعیت (زیرجمعیت‌ها) نیز دارای اهمیت است. از آنجا که این عوامل در ترکیب ژنتیک جمعیت بازتاب می‌یابد، نشانگرهای مولکولی می‌توانند اطلاعات مفیدی را از خصوصیات جمعیتی در این حوزه، فراهم آورند (Avise, 2012؛ Karp *et al.*, 1997؛ Karp *et al.*,1996؛ Lande, 1988؛ Milligan *et al.*, 1994؛ Moritz, 1994 و Vane-Wright *et al.*, 1991)

فناوری نشانگرهای ژنتیکی با روش‌های مبتنی بر فنوتیپ و تعیین آیزوزایم ظاهر شد. این فناوری به سرعت مسیر تکامل را به سمت روش‌های پیچیده مبتنی بر DNA در پیش گرفت. در طی این مسیر همواره برای شناسایی نشانگرهایی که بتوانند تنوع کل ژنوم را پوشش دهند، تلاش شده است (شکل ۱-۱).



شکل ۱-۱- سیر تحول و انواع نشانگرهای مولکولی در طی زمان (گذشته، حال و آینده). اقتباس از Henry, 2013

به طور کلی نشانگرهای مولکولی DNA از لحاظ نوع اطلاعاتی که در یک مکان ژنی حاصل می‌نمایند به سه دسته تقسیم می‌شوند؛ ۱) نشانگرهای غالب دوآللی مانند RAPD و AFLP ۲) نشانگرهای همبارز دوآللی مانند RFLP و ۳) نشانگرهای همبارز چند آللی مانند میکروستلایت‌ها.

محبوبیت نشانگرهای مولکولی در طی زمان از یک نشانگر به نشانگر دیگر تغییر پیدا کرده است. به عنوان مثال اولین تلاش برای ایجاد نقشه ژنتیکی انسانی در مقیاس وسیع، عمدتاً با استفاده از نشانگرهای RFLP صورت گرفت، اما با عمومیت یافتن PCR و تنوع آللی بالایی که در مکان‌های میکروستلایت مشاهده شد، این نشانگرها سریعاً مورد استقبال قرار گرفتند و نقشه‌های ژنتیکی مبتنی بر آنها توسعه یافت (مانند Swinburne ،Maddox *et al.*, 2001 ،Groenen *et al.*, 2000 ،Rohrer *et al.*,1996 ،Kappes *et al.*,1997 Varshney *et al.*, ،Somers *et al.*, 2004 ،Röder *et al.*, 1998 ،Vaiman *et al.*, 1996 ،*et al.*, 2000 Gianfranceschi *et* ،Li *et al.*, 2000 ،McCouch *et al.*,2002 ،Taramino and Tingey, 1996 ،2007 Danin-Poleg *et al.*, و Grando and Frisinghelli, 2015 ،Buhariwalla *et al.*, 2005 ،*al.*, 1998. .(2001

پس از یک دهه حکمرانی نشانگرهای میکروستلایت در حوزه مطالعات ژنومی، نشانگر جدیدی به نام SNP پا به عرصه ظهور گذاشت. از آنجا که SNPها به عنوان نشانگرهای همبارز با تعداد محدودی آلل شناخته می‌شوند ظرفیت ارائه‌ی محدودتری از دامنه اطلاعات ژنتیکی را در مقایسه با نشانگرهای چندآللی میکروستلایت دارا می‌باشند، لذا توسعه این نشانگرها در وهله اول گامی به سمت عقب به نظر می‌رسد. اما آنچه که موجب محبوبیت این نشانگرها شده است نیاز به نشانگرهای ژنتیکی با تراکم بالا برای مطالعه بیماری‌های چندعاملی و پیشرفت‌های اخیر در زمینه شناسایی پلی‌مورفیسم و تکنیک‌های ژنوتیپ‌یابی بوده است (Vignal *et al.*, 2002 ،Donis-Keller *et al.*, 1987 ،Litt and Luty, 1989 ،Weber and May, 1989 ،Murray *et al.*,1994 ،Colette *et al.*,1996 و Weissenbach *et al.*, 1992).

خاطر نشان می‌شود که طراحی صحیح آزمایشات به منظور بهره‌گیری از ظرفیت نشانگرهای مولکولی در حوزه‌های مختلف از اهمیت زیادی برخوردار است. عدم رعایت این امر منجر به دستیابی به نتایج ضعیف و غیرقابل اعتماد خواهد شد.

R و تجزیه ژنتیکی

با ظهور رایانه‌های شخصی و اینترنت، حجم عظیم و رو به تزایدی از داده‌ها در حال تولید می‌باشد. هم اکنون استخراج اطلاعات مفید از میان انبوهی از داده، خود به صنعتی تبدیل شده است. از اینرو چگونگی پردازش داده‌ها در سال‌های اخیر تغییرات شگرفی در برداشته است. از سوی دیگر دانش تحلیل داده‌ها در زمینه‌های مختلف مانند ژنتیک، آمار، روانشناسی، اقتصاد، یادگیری ماشین و غیره، همگام با این فوران اطلاعات، توسعه یافته است. امروزه طیف گسترده‌ای از منابع از قبیل پایگاه‌های داده، فایل‌های متنی، بسته‌های آماری و صفحه‌های گسترده موجود می‌باشد که تحلیلگران داده نیازمند به دسترسی، ادغام، پالایش و پردازش آنها و همچنین ارائه یافته‌هایشان در قالب گزارش‌های تصویری جذاب می‌باشند. برنامه R، دستیابی به این اهداف را به طور جامع مقدور ساخته است.

نرم افزار R، یک زبان برنامه نویسی متن باز[9] و محیطی برای محاسبات آماری است که در سال ۱۹۹۵ در پی دو زبان برنامه نویسی دیگر به نام S و S-Plus، توسط راس ایاکا و رابرت جنتلمن در بخش آمار دانشگاه آکلند نیوزلند ایجاد شد و به سرعت از سوی متخصصان آمار و داده‌کاوی مورد استقبال قرار گرفت (Hornik, 2012؛ R Core Team, 2017؛ Vance, 2009) و محبوبیت آن همچنان رو به افزایش است (Vance, 2009). درحال حاضر گروهی بین‌المللی متشکل از برنامه‌نویسان داوطلب[10]، مسؤلیت پشتیبانی و توسعه این زبان برنامه‌نویسی را برعهده دارند (R Core Team, 2017). از آنجا که این زبان برنامه نویسی متن‌باز است، بطور پیوسته درحال مرور، بهبود و به‌روزرسانی می‌باشد. صدها تابع استاندارد برای مدیریت داده‌ها، تجزیه‌های آماری و رسم نمودار در خود برنامه R، نوشته شده است، اما این امکان نیز فراهم آمده که کاربران، توابع ویژه در زمینه تخصصی خود را ایجاد و در قالب پکیج‌های ضمیمه، عرضه کنند. بدین ترتیب دامنه کاربرد و توانمندی این زبان برنامه‌نویسی روز به روز درحال گسترش می‌باشد. شناخته شده‌ترین قابلیت R، توانایی آن در رسم نمودارهای شکیل، پیچیده و با کیفیت بالا است (Jackman, 2003).

اهمیت پردازش داده‌ها در حوزه ژنتیک جمعیت سبب شده است که نرم‌افزارهای تحلیلی زیادی با این هدف توسعه یابد که از مهمترین آنها می‌توان نرم‌افزارهایSTRUCTURE (Pritchard *et al.*, 2000)، GENETIX (Belkhir *et al.*, 1996-2004)، FSTAT (Goudet, 2002) و Genepop (Raymond and Rousset, 1995) را به عنوان مثال نام برد. با توجه به گسترده بودن موضوعاتی که در ذیل ژنتیک جمعیت قرار می‌گیرند، هر یک از این نرم‌افزارها به ناچار بر جنبه‌های محدودی متمرکز شده‌اند و درنتیجه دارای نقاط قوت و ضعفی می‌باشند. از سوی دیگر مدل‌های ژنتیکی و روش‌های تجزیه ژنتیکی محدود و

[9] Open source
[10] R core-development team

انگشت‌شمار نبوده و روزبه روز درحال تکمیل و توسعه می‌باشند. لذا ضرورت دستیابی به یک برنامه آماری که دارای ظرفیت تحلیل همه جانبه داده‌های ژنتیکی بوده و از سوی دیگر از انعطاف‌پذیری لازم برای سازگاری با مدل‌های ژنتیکی جدیداً توسعه یافته برخوردار باشد، همواره وجود داشته است. ویژگی‌های برنامه R آن را بهترین هدف برای نیل به این مقصود قرار داده است. همانطور که اشاره شد نرم افزار R، یک زبان برنامه نویسی متن باز با امکان افزودن پکیج‌های جانبی توسط کاربران است. این خصوصیت به R توانایی نامحدود در ایجاد توابع برای انواع بی‌شماری از تجزیه‌های ژنتیکی بخشیده و آن را به جامع‌ترین برنامه تجزیه ژنتیکی تبدیل نموده است. به عنوان مثال درحال حاضر R، منحصربه فردترین نرم‌افزار از حیث شمول انواع روش‌های تجزیه چندمتغیره داده‌های ژنتیکی، روش‌های آماری شناسایی الگوهای ژنتیکی-جغرافیایی، امکان شبیه‌سازی انواع داده‌های ژنتیکی و ساختارهای جمعیتی و غیره به شمار می‌رود.

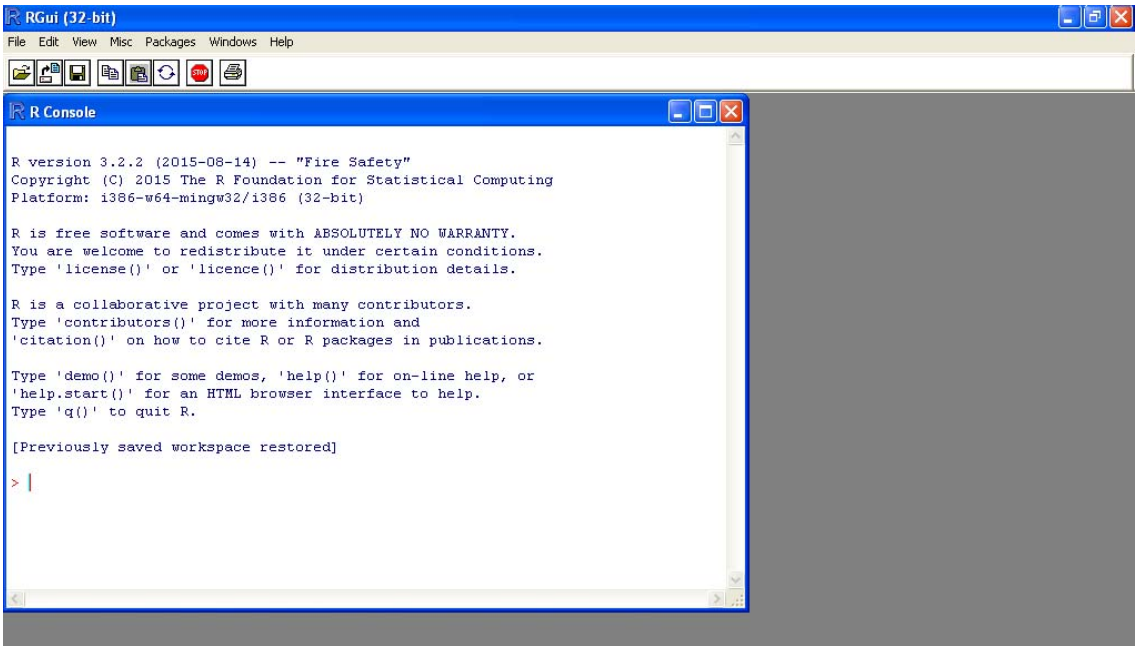
فصل دوم- آشنایی با محیط و انواع متغیرها در R

دسترسی و نصب برنامه R

پایگاه اطلاعاتی اصلی در مورد R، صفحه وب مربوط به پروژه R است (https://www.r-project.org) که حاوی آخرین نسخه نرم‌افزار و همچنین نسخه‌های قدیمی‌تر، راهنمایی‌ها و دستورالعمل‌های لازم برای دریافت برنامه و پکیج‌های همراه و اسناد و منابع از جمله نشریه اختصاصی (The R Journal)، همایش‌ها (موسوم به !UseR) و سایر اخبار و رویدادهای مربوطه می‌باشد. در این پایگاه، نرم‌افزار R به طور رایگان برای نصب در دسترس عموم قرار داده شده است. این نرم افزار بر روی سیستم‌های ویندوز، مک اینتاش و لینوکس قابل نصب و اجرا می‌باشد.

محیط R

پس از اجرای برنامه از طریق منوی استارت (Start menu) یا دوبار کلیک بر روی آیکون مربوط به برنامه (R)، پنجره‌ای بصورت شکل ۲-۱ ظاهر می‌شود که محیط R را نشان می‌دهد.



شکل ۲-۱- محیط برنامه R

در محیط R، دستورات بعد از علامت > درج می‌شوند که توسط خود برنامه ظاهر می‌شود و لزومی به درج آن توسط کاربر نمی‌باشد، لذا علامت > جزو دستور به حساب نمی‌آید، بدین جهت درصورت تکرار این علامت (>>) پیام خطا ظاهر خواهد شد.

R، به املاء صحیح دستورات و همچنین به بزرگی و کوچکی حروف، حساس است. بنابراین برخی پیام‌های خطا ممکن است ناشی از عدم توجه به این امر از سوی کاربر باشد. در R، متغیرهایی که برای ذخیره داده‌ها بکار می‌روند، اشیاء[11] نامیده می‌شوند. در حقیقت یک شیء هر چیزی است که بتوان به آن مقداری منتسب نمود. هر شیء دارای کلاسی است که به R می‌گوید چگونه با آن شیء عمل نماید. تمام اشیاء در طی جلسه کاری در حافظه نگهداری می‌شوند.

نصب و فراخوانی پکیج‌ها

توابع پایه و استاندارد که در خود برنامه R تعریف شده‌اند، بطور پیش فرض در دسترس می‌باشند. اما توابع اختصاصی موجود در پکیج‌ها فقط پس از نصب و فراخوانیِ پکیجِ مربوطه قابل اجرا می‌باشند. لذا برخی از

[11] Objects

پیام‌های خطا ممکن است ناشی از عدم فراخوانیِ پکیج مورد نیاز، باشد. پکیج‌ها، مجموعه‌ای از توابع R می‌باشند که به همراه فایل‌های ضمیمه در پوشه‌ای به نام library، در رایانه شما ذخیره می‌شوند. دستور libPaths(). مسیر مربوط به محل پوشه‌ی library را نشان می‌دهد و تابع library() نشان می‌دهد که چه پکیج‌هایی در این پوشه ذخیره شده است. هنگام نصب R، مجموعه‌ای استاندارد از پکیج‌ها مانند graphics، stats، methods و... همراه با آن، نصب می‌شوندکه حاوی دامنه وسیعی از توابع و مجموعه داده‌ها هستند که بطور پیش‌فرض در دسترس می‌باشند. برای دسترسی به سایر پکیج‌ها لازم است که ابتدا دانلود و نصب و سپس بارگزاری شوند. برای دانلود و نصب پکیج‌ها از دستور install.packages به همراه اسم پکیج در داخل گیومه ("") استفاده می‌شود. به عنوان مثال برای دانلود و نصب پکیج adegenet (پکیج محوری تجزیه ژنتیک جمعیت)، دستور زیر را (بدون علامت > در ابتدای دستور) اجرا کنید:

```
> install.packages("adegenet")
```

با اجرای دستور install.packages فهرستی از سرورها تحت عنوان CRAN mirror sites ظاهر می‌شود. در حقیقت برای سهولت دسترسی و افزایش سرعت دانلود، نسخه مشابهی از پکیج‌ها در سرورهای مختلف قرار داده شده و در این مرحله R از ما می‌خواهد که پایگاهی را انتخاب نماییم که پکیج مورد نظر از آنجا دانلود شود.

همچنین دستور installed.packages() به تنهایی (یعنی بدون درج چیزی در داخل پرانتز)، پکیج‌های نصب شده به همراه مسیر و نسخه آنها را فهرست می‌نماید.

نصب هر پکیج برای یک مرتبه کافی است ولی مانند هر نرم‌افزار دیگری، پکیج‌ها نیز اغلب توسط ایجاد کنندگان‌شان به روز می‌شوند. از دستور update.packages برای به‌روز کردنِ پکیج‌هایی که قبلاً نصب کرده‌اید، استفاده کنید.

پکیج‌های hierfstat، pegas، vegan، mmod، ape، poppr، seqinr، ggplot2 (علاوه بر پکیج adegenet که در بالا شرح داده شد) در این کتاب مورد استفاده قرار خواهند گرفت، لذا لازم است آنها را به شیوه فوق‌الذکر نصب نمایید:

```
> install.packages("adegenet")
> install.packages("pegas")
> install.packages("poppr")
> install.packages("hierfstat")
> install.packages("vegan")
```

```
> install.packages("mmod")
> install.packages("seqinr")
> install.packages("ape")
> install.packages("ggplot2")
```

بدین ترتیب با نصب هر پکیج توسط دستور installed.packages، آن پکیج از پایگاه (CRAN mirror site) انتخابی، دانلود و در پوشه library ذخیره می‌شود. اما برای استفاده از پکیج و محتویات آن، نیاز به فراخوانی توسط دستور library، می‌باشد. به عنوان مثال برای فراخوانی پکیج adegenet باید دستور زیر اجرا شود:

```
> library(adegenet)
        Loading required package: ade4
          /// adegenet 2.0.1 is loaded ////////////
          > overview: '?adegenet'
          > tutorials/doc/questions: 'adegenetWeb()'
          > bug reports/feature requests: adegenetIssues()

        Warning messages:
        1: package ‘adegenet’ was built under R version 3.2.5
        2: package ‘ade4’ was built under R version 3.2.5
```

همانطور که مشاهده می‌شود هنگام اجرای دستور library(adegenet)، نیاز به استفاده از علامت گیومه ("") نمی‌باشد هر چند که استفاده از آن نیز (یعنی بصورت library("adegenet")) ایجاد خطا نمی‌کند.

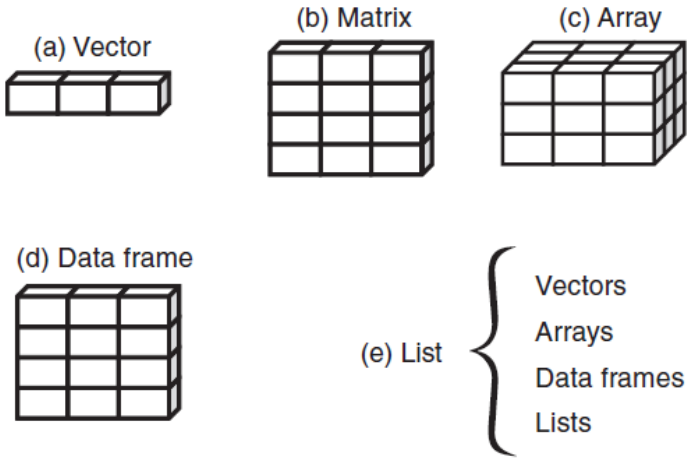
بدیهی است که قبل از اجرای دستور library(adegenet)، بایستی پکیج adegenet، طبق آنچه که در فوق شرح داده شد، از قبل دانلود و نصب شده باشد (وگرنه پیام خطا ظاهر خواهد شد).

با اجرای دستور data() می‌توان فهرستی از مجموعه داده‌های موجود در پکیج‌هایی که فراخوانی شده‌اند را مشاهده نمود:

```
> data()
```

انواع داده‌ها در R

انواع داده‌ها در R، در قالب ساختارهای وکتور، ماتریس، آرایه، قالب داده و لیست دسته‌بندی می‌شوند. درمورد این ساختارهای داده و نحوه پردازش آن‌ها در بخش‌های بعدی توضیح داده خواهد شد.



شکل ۲-۲- انواع داده‌ها در R

برای انتساب مقادیر به اشیاء از علامت <- استفاده می‌شود. هرچند که برای این منظور علامت = را نیز می‌توان بکار برد، ولی روشی استاندارد محسوب نشده و در بین برنامه‌نویسان رایج نیست، مضاف براینکه ممکن است گاهی عمل ننماید.

وکتورها

وکتورها آرایه‌های یک بعدی هستند که می‌توانند در بردارنده داده‌های عدد، کارکتری یا منطقی باشند. از تابع ترکیب کننده c برای تشکیل وکتورها استفاده می‌شود. مثال:

```
> a <- c(3, 7, -2, 3)
> b <- c("one", "two", "three")
> c <- c(TRUE, TRUE, FALSE, TRUE, TRUE, FALSE)
```

با استفاده از علامت : می‌توان یک سری عددی را بصورت وکتور ذخیره نمود:

```
> d <- 3:9
> d
      [1] 3 4 5 6 7 8 9
```

یا

```
> e <- c(5:10)
> e
      [1]  5  6  7  8  9 10
```

با استفاده از علامت [] می‌توان عناصر مورد نظر از داخل یک وکتور را مشاهده یا بطور جداگانه ذخیره نمود. مثلاً برای مشاهده یا ذخیره سومین عنصر از وکتور a:

```
> a[3]
      [1] -2
> f<-a[3]
> f
      [1] -2
```

بسته به ماهیت وکتور (عددی، کاراکتری یا منطقی) می‌توان عملگرها و یا توابع متفاوتی را بر روی آنها اعمال نمود. به عنوان مثال روی وکتورهای عددی، می‌توان اعمال ریاضی انجام داد:

```
> g <- c(1, 3, 5)
> h <- c(1, 2, 1)
> g+h
      [1] 2 5 6
> g-h
      [1] 0 1 4
> g*h
      [1] 1 6 5
> g/h
      [1] 1.0 1.5 5.0
```

ماتریس‌ها

ماتریس‌ها آرایه‌های دو بعدی هستند. برای تشکیل ماتریس‌ها از تابع matrix استفاده می‌کنیم. بدین منظور بایستی از قبل، یک وکتور تعریف کنیم و سپس آنرا به ماتریس تبدیل نماییم. آرگومان nrow (یا استفاده از یک عدد به تنهایی) تعداد ردیف‌های ماتریس را مشخص خواهد نمود. مثال:

```
> a <- c(1:12)
> matrix(a, nrow=2)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    5    7    9   11
[2,]    2    4    6    8   10   12
> matrix(a, 3)
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

همانطور که مشاهده می‌شود اعداد بصورت ستونی در سلول‌های ماتریس قرار می‌گیرند. با استفاده از آرگومان byrow=TRUE می‌توان اعداد را بصورت ردیفی در ماتریس قرار داد:

```
> matrix(a, 3, byrow=TRUE)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

با استفاده از توابع cbind و rbind، نیز می‌توان ماتریس‌ها را تشکیل داد. مثال:

```
> a <- c(1,2,3)
> b <- c(4,5,6)
> c<-cbind(a,b)
> c
     a b
[1,] 1 4
[2,] 2 5
```

```
     [3,] 3 6
> class(c)
     [1] "matrix"
> d<-rbind(a,b)
> d
      [,1] [,2] [,3]
a    1    2    3
b    4    5    6
> class(d)
     [1] "matrix"
```

در دستورات فوق، خروجی تابع class نشان می‌دهد که اشیاء تشکیل شده (c و d) از نوع ماتریس هستند.

با استفاده از علامت [] می‌توان عناصر مورد نظر از داخل یک ماتریس را مشاهده یا بطور جداگانه ذخیره نمود:

```
> d[1,2]
     a
     2
> d[2,3]
     b
     6
```

مشابه با وکتورها، بر روی ماتریس‌ها نیز می‌توان اعمال ریاضی انجام داد. مثال:

```
> 2*c
     a  b
[1,] 2  8
[2,] 4 10
[3,] 6 12
> c-2
     a b
[1,] -1 2
```

```
[2,]  0 3
[3,]  1 4
> c+c
     a  b
[1,] 2  8
[2,] 4 10
[3,] 6 12
```

آرایه‌ها

آرایه‌ها مشابه ماتریس‌ها می‌باشند با این تفاوت که دارای بیش از دو بعد هستند. آرایه‌ها را می‌توان با دستور array تشکیل داد. به عنوان مثال برای چینش اعداد در دو ردیف، سه ستون و چهار گروه خواهیم داشت:

```
> a<-array(1:24, c(2, 3, 4))
> a
, , 1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

, , 2
     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3
     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18

, , 4
     [,1] [,2] [,3]
[1,]   19   21   23
```

```
[2,]   20   22   24
```

```
> class(a)
```

```
[1] "array"
```

قالب داده‌ها

یک قالب داده عبارت از یک آرایه دو بعدی شبیه ماتریس است با این تفاوت که می‌تواند ترکیبی از داده‌های عددی، کاراکتری و منطقی را شامل شود. قالب داده را می‌توان با تابع data.frame تشکیل داد. مثال:

```
> a <- c(1, 7, 5)
> b <- c("Ali", "Mina", "Reza")
>  c <- c(FALSE, TRUE, FALSE)
> d<-data.frame(a,b,c)
> d
```

	a	b	c
1	1	Ali	FALSE
2	7	Mina	TRUE
3	5	Reza	FALSE

```
> class(d)
```

```
[1] "data.frame"
```

با استفاده از علامت [] می‌توان عناصر مورد نظر از داخل یک قالب داده را مشاهده یا بطور جداگانه ذخیره نمود:

```
> d[2,1]
```

```
[1] 7
```

```
> d[3,2]
```

```
[1] Reza
Levels: Ali Mina Reza
```

لیست‌ها

لیست‌ها ترکیبی از انواع مختلف اشیاء هستند. به عبارت دیگر یک لیست می‌تواند حاوی وکتورها، ماتریس‌ها، آرایه‌ها، قالب داده‌ها و یا حتی لیست‌های دیگر باشد. لیست‌ها را می‌توان با استفاده از تابع list تشکیل داد. هر یک از اشیاء (یا اجزاء) تشکیل‌دهنده‌ی لیست را می‌توان با استفاده از علامت [] مشاهده یا ذخیره نمود. مثال:

```
> a <- c(3, -2, 5)
> b <- c("Ali", "Mina", "Reza")
> c<-matrix(7:12, 3)
> d<-data.frame(a,b)
> e<-list(a,b,c,d)
> e[1]
[[1]]
[1]  3 -2  5
> e[2]
[[1]]
[1] "Ali"  "Mina" "Reza"
> e[3]
[[1]]
     [,1] [,2]
[1,]    7   10
[2,]    8   11
[3,]    9   12

> e[4]
[[1]]
  a    b
1 3  Ali
2 -2 Mina
3 5  Reza
```

رسم نمودار

همانطور که در بخش‌های پیشین نیز اشاره شد یکی از توانمندی‌های شناخته شده زبان برنامه‌نویسی R قابلیت آن در ترسیم نمودارهای متنوع است. از آنجا که خروجی بسیاری از توابع تجزیه ژنتیکی بصورت نمودار پراکنش و با استفاده از تابع استاندارد plot قابل نمایش است، در اینجا پیرامون نحوه کار با این تابع و تنظیمات مرتبط، به اختصار توضیح داده می‌شود و جزئیات بیشتر، هنگام استفاده از تابع در زمینه‌ی تجزیه ژنتیکی مورد نظر، ذکر خواهد شد. باید توجه داشت که علاوه بر توابع استاندارد پایه که در خود برنامه R تعبیه شده‌اند، پکیج‌هایی نیز مانند ggplot2 به طور اختصاصی برای رسم نمودارهای شکیل، موجود است که در این زمینه قابل استفاده می‌باشند. نمونه‌ای از کاربرد این پکیج اختصاصی در بخش مربوط به آماره‌های افتراق ژنتیکی جمعیت ارائه خواهد شد.

نمودار ترسیمی توسط تابع plot، متشکل از اجزاء مختلف مانند خط، کاراکتر، برچسب محورها، عنوان‌های اصلی و فرعی می‌باشد که هر یک از این اجزاء، دارای تنظیماتی مانند نوع، اندازه، فونت، رنگ و غیره هستند. در اینجا به عنوان مثال اطلاعات زیر که مربوط به واکنش به دو نوع داروی A و B در بیماران است را به صورت نمودار نمایش می‌دهیم:

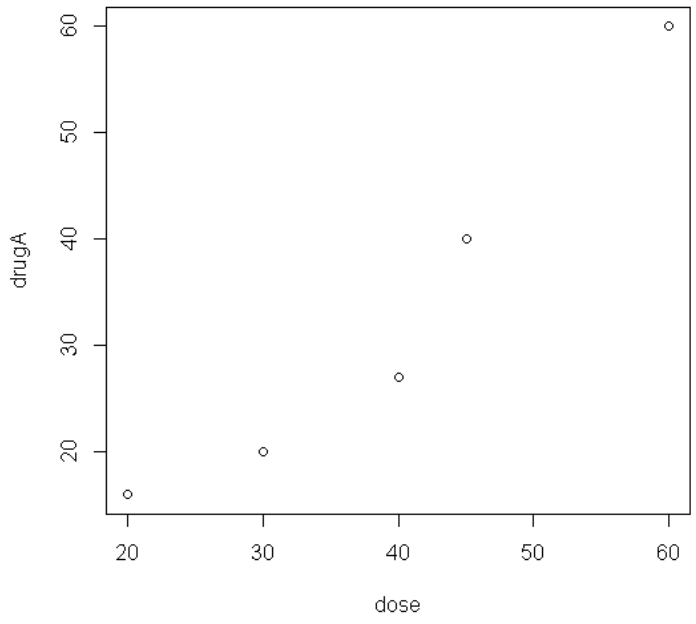
PatientID	Name	Dosage	Response to Drug A	Response to Drug B
PID01	John	20	16	15
PID02	Michael	30	20	18
PID03	George	40	27	25
PID04	Jane	45	40	31
PID05	David	60	60	40

ابتدا باید داده‌های موجود در جدول را به صورت قالب داده ذخیره نماییم:

```
> patientID<-c("PID01","PID02","PID03","PID04","PID05")
> Name<-c("John"," Michael ","George ","Jane","David ")
> dose <-c(20, 30, 40, 45, 60)
> drugA <- c(16, 20, 27, 40, 60)
> drugB <-c(15, 18, 25, 31, 40)
> drugdata <- data.frame(patientID, Name,dose, drugA, drugB)
```

تابع plot، به طور ساده به شکل plot(x,y) اجرا می‌شود که در آن x و y به ترتیب متغیر محورهای افقی و عمودی را تشکیل می‌دهند. به عنوان مثال برای نمایش پاسخ به داروی A، براساس دُز (شکل ۲-۳)، خواهیم داشت:

```
> plot(dose, drugA)
```

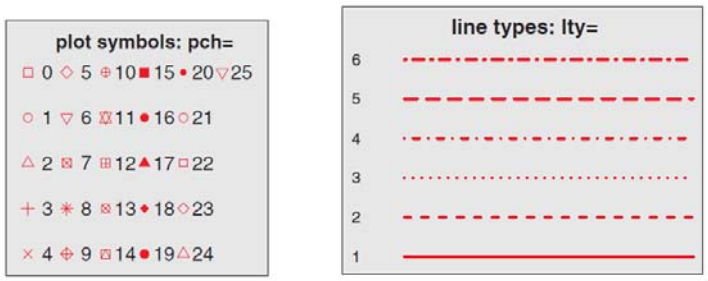


شکل ۲-۳– نمودار ساده واکنش به دُزهای مختلف داروی A در بیماران

نمودار فوق، ساده‌ترین حالتِ نمایشِ پراکنش مشاهدات است. با استفاده از آرگومان‌های مختلف در تابع plot می‌توان نمودار را اصلاح نمود یا اجزاء گوناگونی به آن افزود. آرگومان‌های type="p"، type="l" و type="b" به ترتیب نوع نمودار را اعم از نقطه‌ای، خطی یا هر دو مشخص می‌نمایند. آرگومان pch، برای نمایش نقاط (کاراکترهای) درون نمودار با اشکال متفاوت، استفاده می‌شود. همچنین با آرگومان lty می‌توان نوع خطوط درون نمودار را تعیین نمود. دستورات زیر را به ترتیب اجرا و نتایج را مشاهده نمایید:

```
> plot(dose, drugA, type="p")
> plot(dose, drugA, type="l")
> plot(dose, drugA, type="b")
> plot(dose, drugA, type="b", pch=17)
> plot(dose, drugA, type="b", pch=17, lty=5)
```

همانطور که مشاهده می‌شود کاراکترها و خطوط، با ذکر شماره مشخص می‌شوند که انواع آن در شکل ۲-۴ نشان داده شده است.



شکل ۲-۴– انواع خطوط و علائم (کاراکترها) مورد استفاده در ترسیم نمودار در محیط R

برای تنظیم اندازه کاراکترها و پهنای خطوط به ترتیب از آرگومان‌های cex و lwd استفاده می‌شود:

```
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2)
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2)
```

برای افزودن مشخصات مربوط به هر نقطه (کاراکتر) در نمودار، از تابع جداگانه text استفاده می‌شود. آرگومان‌های cex، pos و col در تابع text به ترتیب اندازه، محل قرار گرفتن و رنگِ مشخصه‌ی افزوده شده به نمودار را تعیین می‌کنند:

```
> plot(dose, drugA, type="b")
> text(dose, drugA,Name, cex=1, pos=2, col="red")
```

برای تنظیم دامنه محورهای افقی و عمودی به ترتیب از آرگومان‌های xlim و ylim استفاده می‌شود:

```
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60))
> text(dose, drugA,Name, cex=1, pos=1, col="red")
```

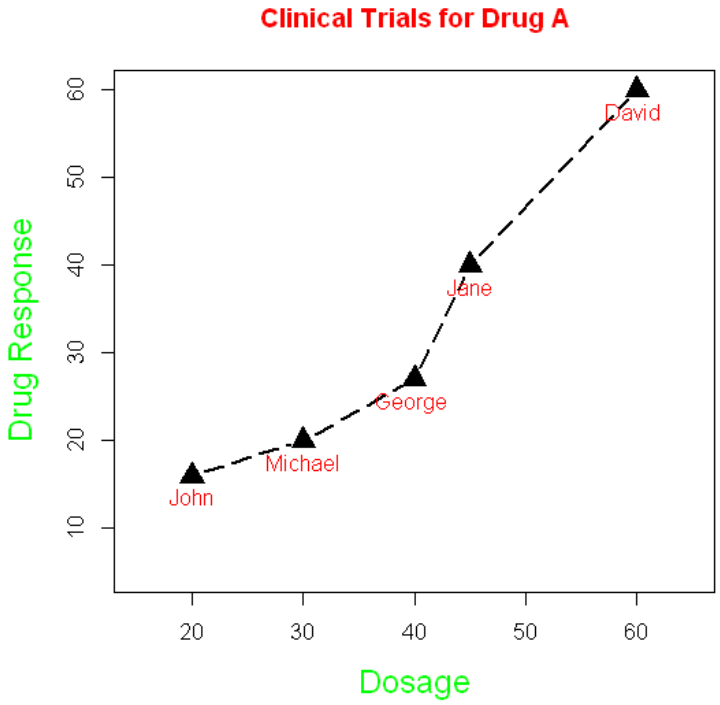
عنوان اصلی نمودار با گزینه main مشخص می‌شود و رنگ و اندازه آن با آرگومان‌های col.main و cex.main، قابل تنظیم است:

```
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60),
main="Clinical Trials for Drug A")
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60),
main="Clinical Trials for Drug A", col.main="red")
```

```
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60), main="Clinical Trials for Drug A", col.main="red", cex.main=2)
```

عنوان اصلی محورهای افقی و عمودی به ترتیب با آرگومان‌های xlab و ylab مشخص می‌شود و اندازه و رنگ آنها به ترتیب با آرگومان‌های cex.lab و col.lab قابل تنظیم است (شکل ۲-۵):

```
> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60), main="Clinical Trials for Drug A", col.main="red", xlab="Dosage", ylab="Drug Response")

> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60), main="Clinical Trials for Drug A", col.main="red", xlab="Dosage", ylab="Drug Response", cex.lab=1.5)

> plot(dose, drugA, type="b", pch=17, lty=5, cex=2, lwd=2, xlim=c(15,65), ylim=c(5,60), main="Clinical Trials for Drug A", col.main="red", xlab="Dosage", ylab="Drug Response", cex.lab=1.5, col.lab="green")

> text(dose, drugA,Name, cex=1, pos=1, col="red")
```



شکل ۲-۵ – نمودار واکنش به دُزهای مختلف داروی A در بیماران

فصل سوم- مدیریت داده‌های ژنتیکی

پکیج‌های R برای تجزیه داده‌های ژنتیکی

به منظور تجزیه داده‌های ژنتیکی، پکیج‌های کارآمدی مانند adegenet، pegas، poppr ، hierfstat، vegan، mmod، ape و غیره طراحی شده است که هریک، دارای کارایی خاصی می‌باشند. این پکیج‌ها حداقل توسط یک مقاله مرجع، معرفی و توسط چندین مقاله دیگر مورد ارجاع قرار گرفته‌اند و لذا اعتبار آنها مورد تأیید جامعه علمی می‌باشد. پکیج adgenet به دلیل اهمیت و ارجاع به آن در زمینه‌های مختلف تجزیه‌های ژنتیکی، در ذیل به اختصار مورد اشاره قرار می‌گیرد.

پکیج adegenet

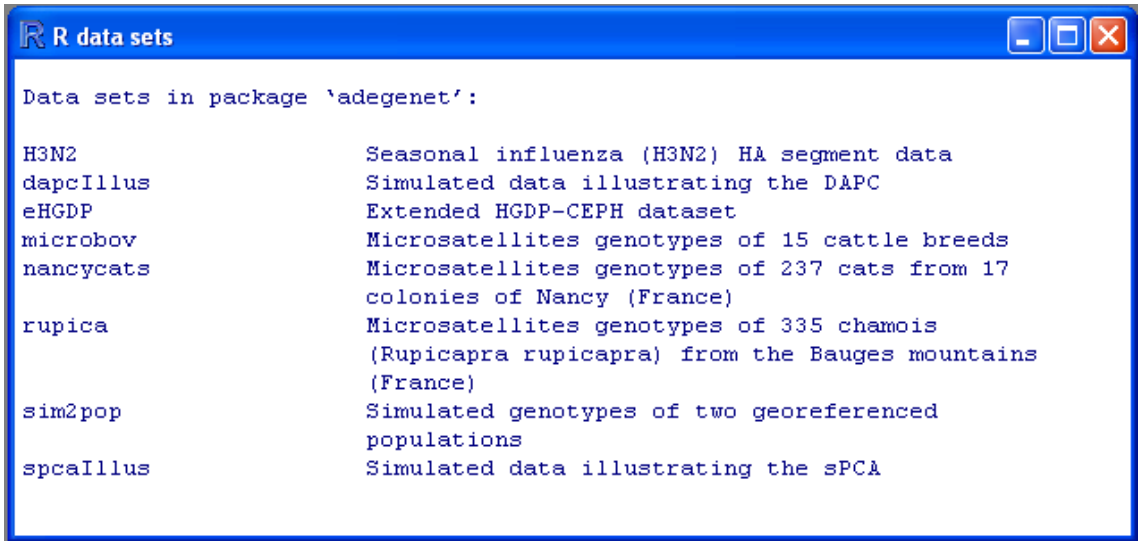
از بین پکیج‌هایی که به منظور تجزیه ژنتیکی توسعه یافته‌اند، پکیج adegenet (Jombart, 2008) را می‌توان به عنوان مجموعه محوری معرفی نمود. این پکیج علاوه بر دارا بودن توابع مفید خصوصاً در زمینه‌ی تجزیه چندمتغیره داده‌های نشانگرهای ژنتیکی، دارای قابلیت تبدیل اشیاء ذیلِ آن، به کلاس‌های مختلف است و بدین ترتیب پل ارتباطی خوبی بین سایر پکیج‌ها ایجاد می‌کند. به عبارت دیگر با استفاده از توابع تعریف شده در این پکیج، به راحتی می‌توان داده‌ها را با فرمت‌های مختلف از سایر نرم‌افزارهای ویژه‌یِ تجزیه ژنتیکی، خوانش و پردازش نمود. همچنین انواع کلاس‌های داده که ذیل پکیج adegenet تشکیل می‌شود به راحتی به یکدیگر و همچنین به کلاس‌ها یا فرمت‌های قابل خوانش توسط پکیج‌ها و نرم‌افزاهای دیگر قابل تبدیل می‌باشد.

نسخه ثابت پکیج adegenet در پایگاه وب http://cran.r-project.org/mirrors.html و نسخه درحال توسعه آن در پایگاه وب http://adegenet.r-forge.r-project.org/ قابل دسترس می‌باشد. هر دو نسخه را می‌توان مستقیماً از درون برنامه R نصب نمود.

در پکیج adegenet، داده‌های ژنتیکی در قالب نشانگرهای همبارز (مانند میکروستلایت)، نشانگرهای غالب به صورت حضور و عدم حضور (مانند AFLP) یا داده‌های حجیم حاصل از توالی‌یابی پربازده، طبقه‌بندی می‌شوند. در این راستا بسته به هدف از تجزیه و نوع داده، سه کلاس از داده‌ها شامل genind (داده‌های افراد)، genpop (داده‌های گروه‌هایی از افراد) و genlight (داده‌های SNP در سطح ژنوم) تعریف شده است.

همچنین در داخل پکیج adegenet، تعدادی مثال در قالب مجموعه داده‌های مختلف به منظور انجام تمرین عملی، فراگیری و درک عملکرد توابعِ موجود، آورده شده است. این مجموعه داده‌ها از قبیل nancycats، microbov، H3N2 و ... می‌باشند که هنگام نصب پکیج adegenet، بطور خودکار، دانلود و با استفاده از تابع data، فراخوانی می‌شوند. برای مشاهده مجموعه داده‌های موجود در پکیج adegenet دستورات زیر را وارد نمایید:

```
> library(adegenet)
> data(package="adegenet")
```

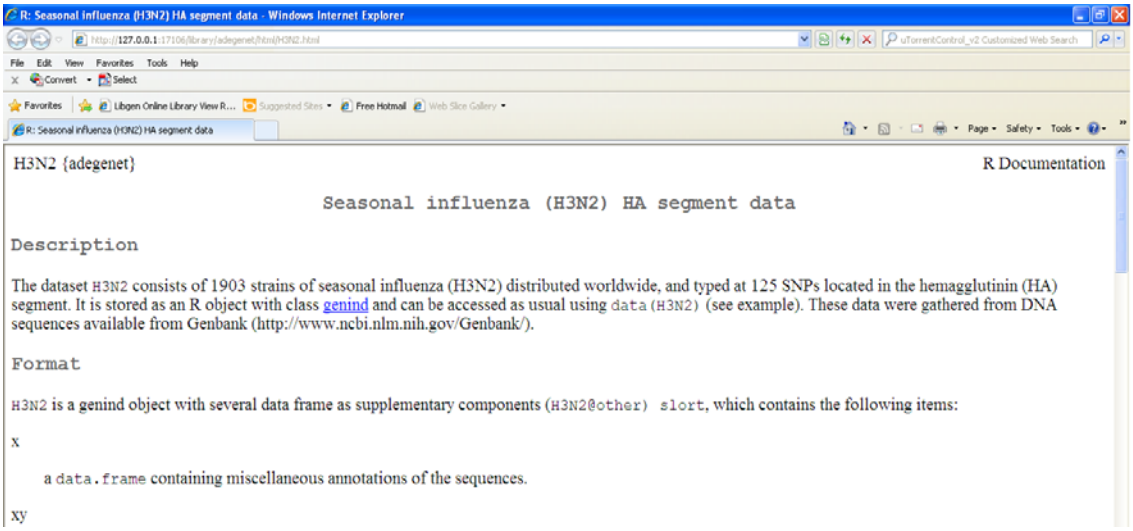


شکل ۳-۱ – نمایش مجموعه داده‌های موجود در پکیج adegenet در محیط R

با اجرای دستوارت فوق، فهرستی ظاهر می‌شود (شکل ۳-۱) که به معرفی مختصر هشت مجموعه داده پرداخته است. به عنوان مثال مجموعه H3N2 شامل داده‌های تعداد صد و بیست و پنج SNP، واقع در قطعه هماگلوتنین از ۱۹۰۳ سوشِ ویروس آنفولانزای فصلی است. برای توضیح بیشتر مربوط به مجموعه داده H3N2، دستور زیر را وارد نمایید:

```
> help(H3N2)
```

همانطور که مشاهده می‌شود صفحه مرورگری باز می‌شود که حاوی اطلاعاتی پیرامون این مجموعه داده و چند مثال از کاربرد برخی توابع است (شکل ۳-۲):



شکل ۳-۲– صفحه مرورگر حاوی توضیحات مربوط به مجموعه داده H3N2

داده‌های ژنتیکی مبتنی بر افراد (اشیاء genind)

یکی از انواع فرمت‌هایی که برای ذخیره داده‌های ژنتیکی استفاده می‌شود، کلاس genind می‌باشد. این کلاس از داده‌ها تحت پکیج adegenet، ایجاد، فراخوانی و پردازش می‌شوند. اشیاء کلاس genind از اجزای مختلفی تشکیل شده‌اند که حاوی اطلاعات گوناگون پیرامون افراد، آلل‌ها، مکان‌های ژنی، جمعیت‌ها، سطح پلوئیدی و غیره است. این اجزاء را می‌توان با استفاده از علامت @، فراخوانی کرد و یا اینکه می‌توان از توابعی که برای این هدف ایجاد شده‌اند، استفاده نمود. برای آشنایی با اشیاء genind و اجزاء آنها، به عنوان مثال مجموعه داده nancycats را فراخوانی می‌نماییم:

```
> library(adegenet)
```

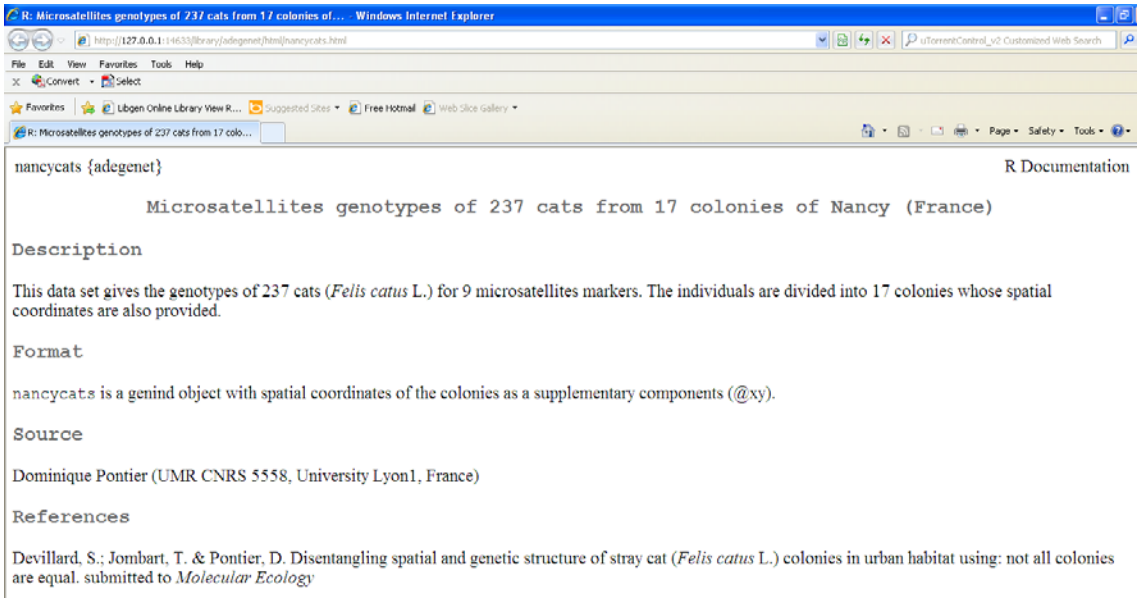
```
> data(nancycats)
```

با استفاده از تابع class می‌توان مشاهده کرد که مجموعه داده nancycats از نوع genind است:

```
> class(nancycats)
    [1] "genind"
```

مجموعه داده nancycats حاوی اطلاعات ۹ نشانگر ریزماهواره برای ۲۳۷ گربه (*Felis catus* L.) در قالب ۱۷ کلونی به همراه مختصات جغرافیایی آنها می‌باشد. برای اطلاعات بیشتر درباره این مجموعه داده، می‌توان از تابع help استفاده نمود که در اینصورت صفحه مرورگری باز خواهد شد که حاوی اطلاعاتی مربوط به این مجموعه، به همراه چند مثال از تجزیه و تحلیل داده‌ها، می‌باشد (شکل ۳-۳):

```
> help(nancycats)
```



شکل ۳-۳– صفحه مرورگر حاوی توضیحات مربوط به مجموعه داده nancycats

با وارد کردن nancycats به عنوان دستور، اجزای مختلف این مجموعه داده، نمایش داده خواهد شد:

```
> nancycats
    /// GENIND OBJECT /////////
     // 237 individuals; 9 loci; 108 alleles; size: 129.4 Kb
     // Basic content
       @tab:  237 x 108 matrix of allele counts
```

@loc.n.all: number of alleles per locus (range: 8-18)

@loc.fac: locus factor for the 108 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: genind(tab = truenames(nancycats)$tab, pop = truenames(nancycats)$pop)

// Optional content

@pop: population of each individual (group size range: 9-23)

@other: a list containing: xy

این اطلاعات را بصورت خلاصه و جمع‌بندی شده با استفاده از تابع summary می‌توان مشاهده نمود:

```
> summary(nancycats)
```

// Number of individuals: 237

// Group sizes: 10 22 12 23 15 11 14 10 9 11 20 14 13 17 11 12 13

// Number of alleles per locus: 16 11 10 9 12 8 12 12 18

// Number of alleles per group: 36 53 50 67 48 56 42 54 43 46 70 52 44 61 42 40 35

// Percentage of missing data: 2.34 %

// Observed heterozygosity: 0.67 0.67 0.68 0.71 0.63 0.57 0.65 0.62 0.45

// Expected heterozygosity: 0.87 0.79 0.8 0.76 0.87 0.69 0.82 0.76 0.61

همانطور که مشاهده می‌شود تعداد ۲۳۷ فرد در قالب ۱۷ گروه، هر یک با اندازه ۱۰، ۲۲، .... ۱۳ فرد، دسته‌بندی شده‌اند. تعداد ۱۰۸ آلل مربوط به ۹ مکان ژنی موجود است و تعداد آلل‌ها به تفکیک مکان‌های ژنی و گروه‌ها ذکر شده است. همچنین میزان ۲/۳۴ درصد داده گمشده وجود دارد که هنگام تجزیه‌های آماری باید مدنظر قرار داد که بعداً به آن پرداخته می‌شود. همچنین داده‌های مربوط به هتروزیگوسیتیِ مشاهده شده و مورد انتظار برای هر مکان ژنی ارائه شده است که در تجزیه و تحلیل‌هایی که بعداً ذکر خواهد شد، مفید خواهد بود.

همانطور که اشاره شد اجزاء مختلف این مجموعه داده را می‌توان با استفاده از علامت @، فراخوانی کرد. مثلاً دستور nancycats@tab داده‌های مربوط به افراد (در ردیف) و آلل‌ها (در ستون) را نمایش خواهد داد که معادل تابع tab(nancycats) می‌باشد :

```
> nancycats@tab
```

یا

```
> tab(nancycats)
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127	fca8.129	fca8.131	fca8.133
N215	NA	NA	NA	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA	NA	NA	NA
N217	0	0	0	0	0	0	0	0
N218	0	0	0	0	0	0	0	1
N219	0	0	0	0	0	0	0	1

…….

با استفاده از تابع head می‌توان اطلاعاتِ آللی شش فرد اول را مشاهده کرد و تصویری کلی از داده‌ها بدست آورد:

```
> head(tab(nancycats))
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127	fca8.129	fca8.131	fca8.133
N215	NA	NA	NA	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA	NA	NA	NA
N217	0	0	0	0	0	0	0	0
N218	0	0	0	0	0	0	0	1
N219	0	0	0	0	0	0	0	1
N220	0	0	0	0	0	0	0	0
	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127	fca8.129	fca8.131	fca8.133
N215	NA	NA	NA	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA	NA	NA	NA
N217	0	0	0	0	0	0	0	0
N218	0	0	0	0	0	0	0	1
N219	0	0	0	0	0	0	0	1
N220	0	0	0	0	0	0	0	0

……..

باید توجه داشت که عددی که در هرستون مشاهده می‌شود، تعداد (شمارش) آلل‌های مشاهده شده مربوط به هر فرد است.

با استفاده از کروشه، می‌توان بخشی از داده‌ها را متمایز نمود، مثلاً دستور tab(nancycats)[1:5,1:5] اطلاعات شمارش پنج آلل اول از پنج فرد اول را نشان می‌دهد:

```
> tab(nancycats)[1:5,1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA
N217	0	0	0	0	0

N218	0	0	0	0	0
N219	0	0	0	0	0

تابع dim ابعاد ماتریسِ تعداد کل افراد و آلل‌ها را مشخص می‌کند:

```
> dim(tab(nancycats))
```

```
[1] 237 108
```

می‌توان با استفاده از تابع as.matrix، ماتریسی متشکل از داده‌های آللی افراد ایجاد، و ردیف و ستون مورد نظر را متمایز و مشاهده نمود:

```
> mat.cats<-as.matrix(tab(nancycats))
> mat.cats[1:4,1:7]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127	fca8.129	fca8.131
N215	NA	NA	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA	NA	NA
N217	0	0	0	0	0	0	0
N218	0	0	0	0	0	0	0

توابع nInd، nLoc، nAll و nPop به ترتیب تعداد افراد، لوکوس‌ها، آلل‌ها (به تفکیک لوکوس‌ها) و جمعیت‌ها را مشخص می‌نماید:

```
> nInd(nancycats)
```

```
[1] 237
```

```
> nLoc(nancycats)
```

```
[1] 9
```

```
> nAll(nancycats)
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
16	11	10	9	12	8	12	12	18

با استفاده از loc.n.all@ نیز می‌توان تعداد آلل‌ها به تفکیک لوکوس‌ها را مشاهده نمود:

```
> nancycats@loc.n.all
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
16	11	10	9	12	8	12	12	18

```
> nPop(nancycats)
[1] 17
```

استفاده از all.names@ لیستی ایجاد می‌کند که در آن اسامی آلل‌ها به تفکیک هر لوکوس، ذکر شده است. با استفاده از علامت $ می‌توان هر یک از اجزای این لیست را فراخوانی و مشاهده نمود. این دستور معادل تابع alleles می‌باشد:

```
> nancycats@all.names
```

یا

```
> alleles(nancycats)
$L1
  01    02    03    04    05    06    07    08    09    10    11    12    13
"117" "119" "121" "123" "127" "129" "131" "133" "135" "137" "139" "141" "143"
  14    15    16
"145" "147" "149"

$L2
  01    02    03    04    05    06    07    08    09    10    11
"128" "130" "132" "136" "138" "140" "142" "144" "146" "148" "150"

$L3
  01    02    03    04    05    06    07    08    09    10
"133" "135" "137" "139" "141" "143" "145" "147" "149" "157"

$L4
  01    02    03    04    05    06    07    08    09
"116" "118" "120" "122" "126" "128" "130" "132" "134"
……
```

```
> nancycats@all.names$L6
  01    02    03    04    05    06    07    08
"138" "140" "142" "144" "146" "148" "150" "152"
```

توابع indNames ، locNames و popNames به ترتیب اسامی افراد، لوکوس‌ها و جمعیت‌ها را مشخص می‌نماید:

```
> indNames(nancycats)
 [1] "N215" "N216" "N217" "N218" "N219" "N220" "N221" "N222" "N223" "N224"
[11] "N7"   "N141" "N142" "N143" "N144" "N145" "N146" "N147" "N148" "N149"
[21] "N151" "N153" "N154" "N155" "N156" "N157" "N158" "N159" "N160" "N161"
[31] "N162" "N163" "N24"  "N25"  "N26"  "N27"  "N28"  "N29"  "N30"  "N31"
.....
> locNames(nancycats)
[1] "fca8"  "fca23" "fca43" "fca45" "fca77" "fca78" "fca90" "fca96" "fca37"
> popNames(nancycats)
 [1] "P01" "P02" "P03" "P04" "P05" "P06" "P07" "P08" "P09" "P10" "P11" "P12"
[13] "P13" "P14" "P15" "P16" "P17"
```

تابع length تعداد عنصر موجود در هر وکتور را برمی‌گرداند، بنابراین در ترکیب با توابع indNames، locNames و popNames، به ترتیب تعداد افراد، لوکوس‌ها و جمعیت‌ها را مشخص می‌نماید و از اینرو معادل توابع nInd، nLoc و nPop می‌باشد:

```
> length(indNames(nancycats))
[1] 237
> length(locNames(nancycats))
[1] 9
> length(popNames(nancycats))
[1] 17
```

از توابع indNames، locNames و popNames همچنین می‌توان برای تغییر نام افراد، لوکوس‌ها و جمعیت‌ها استفاده نمود. به عنوان مثال اگر بخواهیم اسامی افراد در مجموع داده nancycats را تغییر دهیم به شیوه زیر عمل می‌کنیم:

```
> head(indNames(nancycats))
[1] "N215" "N216" "N217" "N218" "N219" "N220"
> indNames(nancycats) <- paste("cat", 1:nInd(nancycats), sep=".")
> head(indNames(nancycats))
```

```
[1] "cat.1" "cat.2" "cat.3" "cat.4" "cat.5" "cat.6"
```

در دستور فوق از تابع paste، برای نام‌گذاری افراد استفاده شد. این تابع برای چسباندن کاراکترها به همدیگر بکار می‌رود. در دستور فوق، با استفاده از آرگومان 1:nInd(nancycats)، شماره ۱ الی ۲۳۷، (که همان nInd(nancycats) است) به عبارت "cat" چسبانده شده و بین آنها از کاراکتر جداکننده "." استفاده می‌شود. به عنوان مثال اگر بخواهیم یک وکتور کاراکتری بصورت ("Gene1" "Gene2" … "Gene5") یا ("Gene.1" "Gene.2" …. "Gene.5") ایجاد کنیم، می‌توانیم، دستورات زیر را اجرا کنیم:

```
> x<- paste("Gene", 1:5, sep="")
> x
      [1] "Gene1" "Gene2" "Gene3" "Gene4" "Gene5"
> y<- paste("Gene", 1:5, sep=".")
> y
       [1] "Gene.1" "Gene.2" "Gene.3" "Gene.4" "Gene.5"
> class(y)
      [1] "character"
```

جداسازی بخشی از داده‌ها

جدا کردن انتخابی لوکوس‌ها

می‌توان بخشی از داده‌ها را مجزا نمود و در قالب شیء genind بطور جداگانه ذخیره نمود. به عنوان مثال در مجموعه داده nancycats لوکوس‌های fca45 و fca78 به شیوه زیر قابل استخراج و ذخیره جداگانه می‌باشند:

```
> library(adegenet)
> data(nancycats)
> locNames(nancycats)
      [1] "fca8"  "fca23" "fca43" "fca45" "fca77" "fca78" "fca90" "fca96" "fca37"
> loc.cats<-nancycats[loc=c("fca45","fca78")]
```

```
> loc.cats
        /// GENIND OBJECT /////////
         // 237 individuals; 2 loci; 17 alleles; size: 32 Kb
         // Basic content
           @tab:  237 x 17 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 8-9)
           @loc.fac: locus factor for the 17 columns of @tab
           @all.names: list of allele names for each locus
           @ploidy: ploidy of each individual  (range: 2-2)
           @type:  codom
           @call: .local(x = x, i = i, j = j, loc = ..1, drop = drop)

         // Optional content
           @pop: population of each individual (group size range: 9-23)
           @other: a list containing: xy

> locNames(loc.cats)
        [1] "fca45" "fca78"
```

برای جداکردن اطلاعات لوکوس‌ها از یکدیگر همچنین می‌توان از تابع seploc از پکیج adegenet استفاده نمود. داده‌های nancycats شامل ۹ لوکوس می‌باشند، می‌توان با استفاده از تابع seploc هر یک از ۹ لوکوس مذکور را در قالب یک شیء genind جداگانه ذخیره نمود. مثال:

```
>  nLoc(nancycats)
        [1] 9
> loc.cats <- seploc(nancycats)
>  class(loc.cats)
        [1] "list"
```

همانطور که مشاهده می‌شود شیء ایجاد شده (loc.cats) از نوع لیست است. این لیست متشکل از ۹ جزء است که هر جزء یک شیء genind، مربوط به اطلاعات یکی از لوکوس‌های داده‌های nancycats است:

```
> names(loc.cats)
    [1] "fca8"  "fca23" "fca43" "fca45" "fca77" "fca78" "fca90" "fca96" "fca37"
> length(names(loc.cats))
    [1] 9
> loc.cats [[1]]
    /// GENIND OBJECT /////////
     // 237 individuals; 1 locus; 16 alleles; size: 30.5 Kb
     // Basic content
       @tab:  237 x 16 matrix of allele counts
       @loc.n.all: number of alleles per locus (range: 16-16)
       @loc.fac: locus factor for the 16 columns of @tab
       @all.names: list of allele names for each locus
       @ploidy: ploidy of each individual  (range: 2-2)
       @type:  codom
       @call: .local(x = x)
     // Optional content
       @pop: population of each individual (group size range: 9-23)
       @other: a list containing: xy
```

```
> tab(loc.cats [[1]])[1:5,1:5]
     fca8.117 fca8.119 fca8.121 fca8.123 fca8.127
N215       NA       NA       NA       NA       NA
N216       NA       NA       NA       NA       NA
N217        0        0        0        0        0
N218        0        0        0        0        0
N219        0        0        0        0        0
```

جدا کردن انتخابی جمعیت‌ها

بطور مشابه با لوکوس‌ها، می‌توان داده‌های جمعیت‌های خاصی را مجزا و در قالب شیء genind بطور جداگانه ذخیره نمود. در مثال زیر داده‌های جمعیت‌های ششم و پانزدهم از مجموعه nancycats (که مجموعاً ۲۲ فرد را شامل می‌شود)، در قالب شیء genind بطور جداگانه (با نام p.cats.6.15) ذخیره می‌شود:

```
> library(adegenet)
> data(nancycats)
> popNames(nancycats)
 [1] "P01" "P02" "P03" "P04" "P05" "P06" "P07" "P08" "P09" "P10" "P11" "P12"
[13] "P13" "P14" "P15" "P16" "P17"
> p.cats.6.15 <- nancycats[pop=c(6,15)]
> p.cats.6.15
/// GENIND OBJECT /////////
 // 22 individuals; 9 loci; 108 alleles; size: 24.3 Kb
 // Basic content
   @tab:  22 x 108 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 8-18)
   @loc.fac: locus factor for the 108 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 2-2)
   @type:  codom
   @call: .local(x = x, i = i, j = j, pop = ..1, drop = drop)
 // Optional content
   @pop: population of each individual (group size range: 11-11)
   @other: a list containing: xy

> popNames(p.cats.6.15)
[1] "P06" "P15"
> pop(p.cats.6.15)
 [1] P06 P06 P06 P06 P06 P06 P06 P06 P06 P06 P06 P15 P15 P15 P15 P15 P15 P15 P15
```

[20] P15 P15 P15

Levels: P06 P15

همچنین با استفاده تابع popsub از پکیج poppr، می‌توان زیرمجموعه‌ای از جمعیت‌ها تشکیل داد. به عنوان مثال مجموعه microbov شامل داده‌های ۳۰ نشانگر میکروستلایت برای ۷۰۴ رأس گاو می‌باشد (Laloe *et al.*, 2007). چنانچه بخواهیم زیر مجموعه‌ای از داده‌های micobov با جمعیت‌های دارای اندازه ۵۰ فرد با استفاده از تابع popsub تشکیل دهیم، دستورات زیر را اجرا می‌نماییم:

```
> library(adegenet)
> library(poppr)
> data(microbov)
> table(pop(microbov))
```

Borgou	Zebu	Lagunaire	NDama	Somba
50	50	51	30	50
Aubrac	Bazadais	BlondeAquitaine	BretPieNoire	Charolais
50	47	61	31	55
Gascon	Limousin	MaineAnjou	Montbeliard	Salers
50	50	49	30	50

```
> P50.mic <- popsub(microbov, sublist=c(1:6, 11:15), blacklist=c(3,4,13,14))
```

در تابع فوق، آرگومان sublist، دامنه‌ای از جمعیت‌های اول تا ششم و همچنین یازدهم تا پانزدهم را انتخاب می‌کند و آرگومان blacklist، جمعیت‌های سوم، چهارم، سیزدهم و چهاردهم را از این دامنه حذف می‌کند. بنابراین شیء ایجاد شده از نوع genind و مشتمل بر جمعیت‌های اول، دوم، پنجم، ششم، یازدهم، دوازدهم و پانزدهم (یعنی جمعیت‌های با اندازه ۵۰ فرد) از مجموعه داده microbov خواهد بود:

```
> P50.mic
/// GENIND OBJECT /////////
 // 350 individuals; 30 loci; 351 alleles; size: 539.3 Kb
 // Basic content
   @tab:  350 x 351 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 5-20)
   @loc.fac: locus factor for the 351 columns of @tab
   @all.names: list of allele names for each locus
```

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: popsub(gid = microbov, sublist = c(1:6, 11:15), blacklist = c(3,

4, 13, 14))

// Optional content

@pop: population of each individual (group size range: 50-50)

@other: a list containing: coun breed spe

```
> table(pop(P50.mic))
```

Borgou	Zebu	Somba	Aubrac	Gascon	Limousin	Salers
50	50	50	50	50	50	50

با استفاده از دستور زیر ده جمعیت اول از داده‌های microbov، بجز Bazadais انتخاب می‌شوند:

```
> Pf10.mic <- popsub(microbov, sublist=1:10, blacklist="Bazadais")
> table(pop(Pf10.mic))
```

Borgou	Zebu	Lagunaire	NDama	Somba
50	50	51	30	50
Aubrac	BlondeAquitaine	BretPieNoire	Charolais	
50	61	31	55	

جدا کردن جمعیت‌ها برحسب اندازه

تابع selPopSize از پکیج adegenet، اندازه نمونه (تعداد افراد) در جمعیت‌های یک شیء genind را بررسی می‌کند و فقط داده‌های جمعیت‌هایی را حفظ می‌کند که اندازه آن‌ها از میزان حداقلی که مشخص شده بیشتر باشد.

مثال: چنانچه بخواهیم در مجموعه داده nancycats فقط جمعیت‌هایی حفظ شود که دارای اندازه حداقل ۱۵ فرد باشد، دستورات زیر را اجرا می‌نماییم :

```
> library(adegenet)
>  data(nancycats)
>  nPop(nancycats)
```

```
[1] 17
```

```
> table(pop(nancycats))
```

P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	P16	P17
10	22	12	23	15	11	14	10	9	11	20	14	13	17	11	12	13

همانطور که مشاهده می‌شود مجموعه داده nancycats از ۱۷ جمعیت با اندازه‌های متفاوت، تشکیل شده است. قصد داریم جمعیت‌هایی که دارای اندازه کوچکتر از ۱۵ فرد می‌باشند را از مجموعه داده‌ها، حذف نماییم:

```
> cat.pop.size15 <- selPopSize(nancycats, n=15)
```

```
> cat.pop.size15
```

```
/// GENIND OBJECT /////////

 // 97 individuals; 9 loci; 108 alleles; size: 59.1 Kb

 // Basic content

   @tab:  97 x 108 matrix of allele counts

   @loc.n.all: number of alleles per locus (range: 8-18)

   @loc.fac: locus factor for the 108 columns of @tab

   @all.names: list of allele names for each locus

   @ploidy: ploidy of each individual  (range: 2-2)

   @type:  codom

   @call: .local(x = x, i = i, j = j, drop = drop)

 // Optional content

   @pop: population of each individual (group size range: 15-23)

   @other: a list containing: xy
```

```
> nPop(cat.pop.size15)
```

```
[1] 5
```

```
> table(pop(cat.pop.size15))
```

P02	P04	P05	P11	P14
22	23	15	20	17

همانطور که مشاهده می‌شود، شیء جدید از نوع genind و متشکل از پنج جمعیت با اندازه ۱۵ فرد یا بیشتر است.

ادغام داده‌ها

با استفاده از تابع repool می‌توان ژنوتیپ‌های اشیاء genind مختلف را با هم ادغام کرد و مجموعاً یک شیء genind واحد تشکیل داد.

مثال: می‌خواهیم زیرمجموعه‌ای از داده‌های microbov تهیه کنیم که فقط شامل جمیعت‌های Lagunaire و Gascon باشد. برای اینکار ابتدا با استفاده از تابع seppop، دو جمعیت مذکور را از مجموعه داده microbov بطور جداگانه استخراج می‌کنیم و سپس توسط تابع repool در هم ادغام می‌نماییم.

```
> library(adegenet)
> data(microbov)
> names(seppop(microbov))
 [1] "Borgou"         "Zebu"           "Lagunaire"      "NDama"
 [5] "Somba"          "Aubrac"         "Bazadais"       "BlondeAquitaine"
 [9] "BretPieNoire"   "Charolais"      "Gascon"         "Limousin"
[13] "MaineAnjou"     "Montbeliard"    "Salers"
> p1<-seppop(microbov)$Lagunaire
> nInd(p1)
[1] 51
> p2<-seppop(microbov)$Gascon
> nInd(p2)
[1] 50
> p.1.2<-repool(p1, p2)
> nInd(p.1.2)
[1] 101
```

همانطور که مشاهده می‌شود جمعیت‌های Lagunaire و Gascon به ترتیب دارای ۵۱ و ۵۰ عضو بودند و مجموعه داده جدید (p.1.2) شامل ۱۰۱ عضو می‌باشد

انتساب جمعیت

از تابع setPop می‌توان برای اختصاص دادن خصوصیات جمعیتی به یک شیء genind که از قبل موجود است و یا ایجاد یک شیء genind جدید استفاده کرد.

مثال: مجموعه microbov شامل داده‌های ۳۰ نشانگر میکروستلایت برای ۷۰۴ رأس گاو می‌باشد (Laloe *et al.*, 2007). در بخش other از این مجموعه داده، سه لایه (سه نوع دسته‌بندی) موجود است، که شامل کشور (coun)، نژاد (breed) و گونه (spe) می‌باشند:

```
> library(adegenet)
> data(microbov)
> microbov
    /// GENIND OBJECT /////////
     // 704 individuals; 30 loci; 373 alleles; size: 1.1 Mb
     // Basic content
       @tab:  704 x 373 matrix of allele counts
       @loc.n.all: number of alleles per locus (range: 5-22)
       @loc.fac: locus factor for the 373 columns of @tab
       @all.names: list of allele names for each locus
       @ploidy: ploidy of each individual  (range: 2-2)
       @type:  codom
       @call: genind(tab = truenames(microbov)$tab, pop = truenames(microbov)$pop)
     // Optional content
       @pop: population of each individual (group size range: 30-61)
       @other: a list containing: coun  breed  spe
```

برای مشاهده داده‌های مربوط به دسته‌بندی‌های مذکور (یعنی کشور (coun)، نژاد (breed) و گونه (spe)) می‌توان دستورات زیر را اجرا نمود:

```
> other(microbov)$coun
[1] AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF
[26] AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF AF
```

…….

Levels: AF FR

> other(microbov)$breed

[1] Borgou Borgou Borgou Borgou
[5] Borgou Borgou Borgou Borgou
…….

15 Levels: Aubrac Bazadais BlondeAquitaine Borgou BretPieNoire ... Zebu

> other(microbov)$spe

[1] BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI

[26] BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI

…….

Levels: BI BT

اما این دسته‌بندی‌ها هنوز به ژنوتیپ‌های موجود در شیء genind منتسب نشده‌اند و فقط به عنوان داده‌های تکمیلی در شیء genind موجود می‌باشند. انتساب یا عدم انتساب دسته‌بندی به ژنوتیپ‌ها در یک شیء genind را می‌توان با استفاده از تابع strata بررسی نمود:

> strata(microbov)

NULL

باید توجه داشت که عمل دسته‌بندی (لایه‌بندی) ژنوتیپ‌ها، با عمل انتساب جمعیت به آنها، متفاوت است. در همین مجموعه داده microbov (که براساس دستور فوق مشخص شد هنوز فاقد دسته‌بندی می‌باشد)، ژنوتیپ‌ها براساس جمعیت، از یکدیگر تفکیک شده‌اند. این موضوع را می‌توان با تابع pop بررسی نمود:

> pop(microbov)

[1] Borgou Borgou Borgou Borgou
[5] Borgou Borgou Borgou Borgou
[9] Borgou Borgou Borgou Borgou
[13] Borgou Borgou Borgou Borgou
[17] Borgou Borgou Borgou Borgou
…….

با استفاده از تابع strata می‌توان دسته‌بندی را به ژنوتیپ‌ها منتسب نمود:

> strata(microbov) <- data.frame(other(microbov))

```
> strata(microbov)
```

	coun	breed	spe
1	AF	Borgou	BI
2	AF	Borgou	BI
3	AF	Borgou	BI
4	AF	Borgou	BI
5	AF	Borgou	BI

باید توجه داشت بعد از تخصیص لایه‌ها به ژنوتیپ‌ها با دستور فوق، کماکان دسته‌بندی‌های مذکور به عنوان جمعیت‌های تفکیک کننده‌ی ژنوتیپ‌ها محسوب نمی‌شوند. با اجرای مجدد دستور pop می‌توان صحت این امر را بررسی نمود:

```
> pop(microbov)
 [1] Borgou  Borgou  Borgou  Borgou
 [5] Borgou  Borgou  Borgou  Borgou
 [9] Borgou  Borgou  Borgou  Borgou
[13] Borgou  Borgou  Borgou  Borgou
……..
```

به منظور اینکه دسته‌بندی‌های تخصیص یافته به ژنوتیپ‌ها به عنوان جمعیت‌های تفکیک کننده‌ی ژنوتیپ‌ها محسوب شوند، می‌توان از تابع setPop استفاده نمود. به عنوان مثال چنانچه بخواهیم ژنوتیپ‌ها را بصورت سلسله مراتبی برحسب کشور (coun) و سپس، نژاد (breed) دسته بندی کنیم، دستور زیر را اجرا می‌نماییم:

```
> setPop(microbov) <- ~coun/breed
> pop(microbov)
 [1] AF_Borgou  AF_Borgou  AF_Borgou
 [4] AF_Borgou  AF_Borgou  AF_Borgou
 [7] AF_Borgou  AF_Borgou  AF_Borgou
[10] AF_Borgou  AF_Borgou  AF_Borgou
[13] AF_Borgou  AF_Borgou  AF_Borgou
[16] AF_Borgou  AF_Borgou  AF_Borgou
[19] AF_Borgou  AF_Borgou  AF_Borgou
……
[697] FR_Salers  FR_Salers  FR_Salers
```

```
[700] FR_Salers        FR_Salers        FR_Salers
[703] FR_Salers        FR_Salers
15 Levels: AF_Borgou AF_Zebu AF_Lagunaire AF_NDama AF_Somba ... FR_Salers
```

```
> names(seppop(microbov))
 [1] "AF_Borgou"         "AF_Zebu"           "AF_Lagunaire"
 [4] "AF_NDama"          "AF_Somba"          "FR_Aubrac"
 [7] "FR_Bazadais"       "FR_BlondeAquitaine" "FR_BretPieNoire"
[10] "FR_Charolais"      "FR_Gascon"         "FR_Limousin"
[13] "FR_MaineAnjou"     "FR_Montbeliard"    "FR_Salers"
```

اکنون می‌توان افرادی که به کشور و نژاد خاصی تعلق دارند را در قالب یک شیء genind مجزا، تفکیک نمود:

```
> seppop(microbov)$FR_BlondeAquitaine
 /// GENIND OBJECT /////////
 // 61 individuals; 30 loci; 373 alleles; size: 134 Kb
 // Basic content
   @tab:  61 x 373 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 5-22)
   @loc.fac: locus factor for the 373 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 2-2)
   @type:  codom
   @call: .local(x = x, i = i, j = j, treatOther = ..1, quiet = ..2, drop = drop)
 // Optional content
   @pop: population of each individual (group size range: 61-61)
   @strata: a data frame with 3 columns ( coun, breed, spe )
   @other: a list containing: coun  breed  spe
```

براساس نتایج فوق، ۶۱ فرد به کشور فرانسه (FR) و نژاد BlondeAquitaine تعلق دارند که اطلاعات مربوط به آنها در قالب یک شیء genind مجزا، ذخیره شده است.

همچنین اگر بخواهیم ژنوتیپ‌ها را بصورت سلسله مراتبی برحسب کشور (coun) و سپس براساس گونه (spe) دسته‌بندی کنیم، می‌توانیم دستور زیر را اجرا می‌نماییم:

```
> setPop(microbov) <- ~coun/spe
> pop(microbov)
    [1] AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI
   [13] AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI
   [25] AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI AF_BI
   ……..
  [685] FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT
  [697] FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT FR_BT
  Levels: AF_BI AF_BT FR_BT
```

با استفاده از تابع setPop همچنین می‌توان مستقیماً شیء genind جدید ایجاد نمود. مثلاً چنانچه بخواهیم ژنوتیپ‌ها را براساس گونه (spe) به عنوان تنها مشخصه جمعیتی، در قالب شیء genind جدید، متمایز نماییم، می‌توانیم از دستور زیر استفاده کنیم:

```
> spe.microbov <- setPop(microbov, ~spe)
> spe.microbov
  /// GENIND OBJECT /////////
   // 704 individuals; 30 loci; 373 alleles; size: 1.1 Mb
   // Basic content
     @tab:  704 x 373 matrix of allele counts
     @loc.n.all: number of alleles per locus (range: 5-22)
     @loc.fac: locus factor for the 373 columns of @tab
     @all.names: list of allele names for each locus
     @ploidy: ploidy of each individual  (range: 2-2)
     @type:  codom
     @call: genind(tab = truenames(microbov)$tab, pop = truenames(microbov)$pop)
   // Optional content
     @pop: population of each individual (group size range: 100-604)
     @strata: a data frame with 3 columns ( coun, breed, spe )
```

@other: a list containing: coun breed spe

مشاهده می‌شود که در شیء genind جدید (spe.microbov) کماکان اطلاعات مربوط به کشور (coun)، نژاد (breed) و گونه (spe)، موجود می‌باشد. اما معیار تفکیک جمعیتی افراد، برحسب گونه (spe) است:

```
> pop(spe.microbov)
 [1] BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI
[26] BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI
[51] BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI BI
 ……
[701] BT BT BT BT
Levels: BI BT
```

داده‌های ژنتیکی مبتنی بر جمعیت‌ها (اشیاء genpop)

در پکیج adegenet علاوه بر کلاس genind برای داده‌های انفرادی، کلاس دیگری از داده‌ها به نام genpop به منظور ذخیره و پردازش در قالب جمعیت‌ها (گروه‌های مشخصی از افراد) تعریف شده است. اشیاء کلاس genind را می‌توان به آسانی با دستور genind2genpop، به اشیاء کلاس genpop تبدیل نمود:

```
> library(adegenet)
> p.cats<-genind2genpop(nancycats)
 Converting data from a genind to a genpop object...
 ...done.
> p.cats
 /// GENPOP OBJECT /////////
  // 17 populations; 9 loci; 108 alleles; size: 21 Kb
  // Basic content
    @tab:  17 x 108 matrix of allele counts
    @loc.n.all: number of alleles per locus (range: 8-18)
    @loc.fac: locus factor for the 108 columns of @tab
    @all.names: list of allele names for each locus
    @ploidy: ploidy of each individual  (range: 2-2)
```

@type: codom

@call: genind2genpop(x = nancycats)

// Optional content

@other: a list containing: xy

مشابه با اشیاء کلاس genind، اشیاء کلاس genpop نیز از اجزای مختلفی تشکیل شده‌اند که حاوی اطلاعات گوناگون پیرامون جمعیت‌ها، آلل‌ها، لوکوس‌ها، سطح پلوئیدی و غیره است که به همان صورت، این اجزاء را می‌توان با استفاده از علامت @، فراخوانی کرد و یا اینکه می‌توان از توابعی که برای این هدف ایجاد شده‌اند، استفاده نمود. به دلیل مشابهت نحوه فراخوانی اجزاء تشکیل دهنده‌ی شیء genpop با شیء genind و یا توابعی که بدین منظور ایجاد شده‌اند، از شرح تک تک این موارد خودداری می‌کنیم و فقط به ذکر چند مثال پیرامون مجموعه داده nancycats اکتفا می‌نماییم:

```
> names(p.cats)
```

[1] "tab" "loc.fac" "loc.n.all" "all.names" "ploidy" "type"

[7] "other" "call"

```
> str(p.cats)
```

Formal class 'genpop' [package "adegenet"] with 8 slots

..@ tab : int [1:17, 1:108] 0 0 0 0 0 0 0 0 0 0 ...

.. ..- attr(*, "dimnames")=List of 2

.. .. ..$: chr [1:17] "P01" "P02" "P03" "P04" ...

.. .. ..$: chr [1:108] "fca8.117" "fca8.119" "fca8.121" "fca8.123" ...

..@ loc.fac : Factor w/ 9 levels "fca8","fca23",..: 1 1 1 1 1 1 1 1 1 1 ...

..@ loc.n.all: Named int [1:9] 16 11 10 9 12 8 12 12 18

.. ..- attr(*, "names")= chr [1:9] "fca8" "fca23" "fca43" "fca45" ...

..@ all.names:List of 9

.. ..$ fca8 : chr [1:16] "117" "119" "121" "123" ...

.. ..$ fca23: chr [1:11] "128" "130" "132" "136" ...

.. ..$ fca43: chr [1:10] "133" "135" "137" "139" ...

.. ..$ fca45: chr [1:9] "116" "118" "120" "122" ...

.. ..$ fca77: chr [1:12] "132" "142" "144" "146" ...

.. ..$ fca78: chr [1:8] "138" "140" "142" "144" ...

.. ..$ fca90: chr [1:12] "181" "185" "187" "189" ...

.. ..$ fca96: chr [1:12] "091" "101" "103" "105" ...

.. ..$ fca37: chr [1:18] "182" "184" "186" "192" ...

..@ ploidy   : int 2

..@ type   : chr "codom"

..@ other   :List of 1

.. ..$ xy: num [1:17, 1:2] 263 184 391 459 183 ...

.. .. ..- attr(*, "dimnames")=List of 2

.. ..$: chr [1:17] "P01" "P02" "P03" "P04" ...

.. ..$: chr [1:2] "x" "y"

..@ call   : language genind2genpop(x = nancycats)

> tab(p.cats)[1:5,1:5]

یا

> p.cats@tab[1:5,1:5]

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
P01	0	0	0	0	0
P02	0	0	0	0	0
P03	0	0	0	4	0
P04	0	0	0	3	0
P05	0	0	0	1	0

همانطور که در فوق مشاهده می‌شود، در اشیاء genpop بجای افراد، اسامی جمعیت‌ها جایگزین شده است.

> summary(p.cats)

// Number of populations: 17

// Number of alleles per locus: 16 11 10 9 12 8 12 12 18

// Number of alleles per group: 36 53 50 67 48 56 42 54 43 46 70 52 44 61 42 40 35

// Percentage of missing data: 0 %

> nPop(p.cats)

[1] 17

> nLoc(p.cats)

```
[1] 9
> nAll(p.cats)
```

یا

```
> p.cats@loc.n.all
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
16	11	10	9	12	8	12	12	18

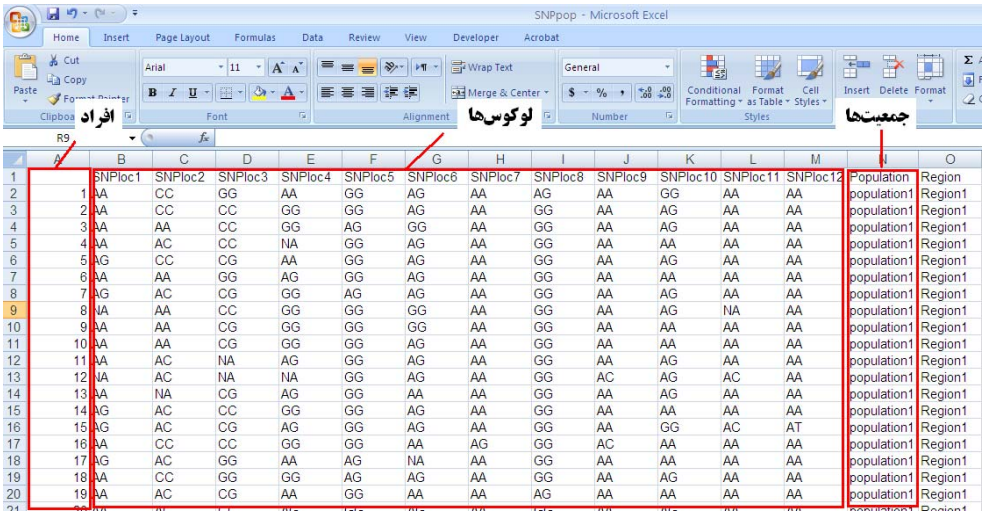
و الی آخر....

نحوه تنظیم و خوانش فایل داده‌های شخصی

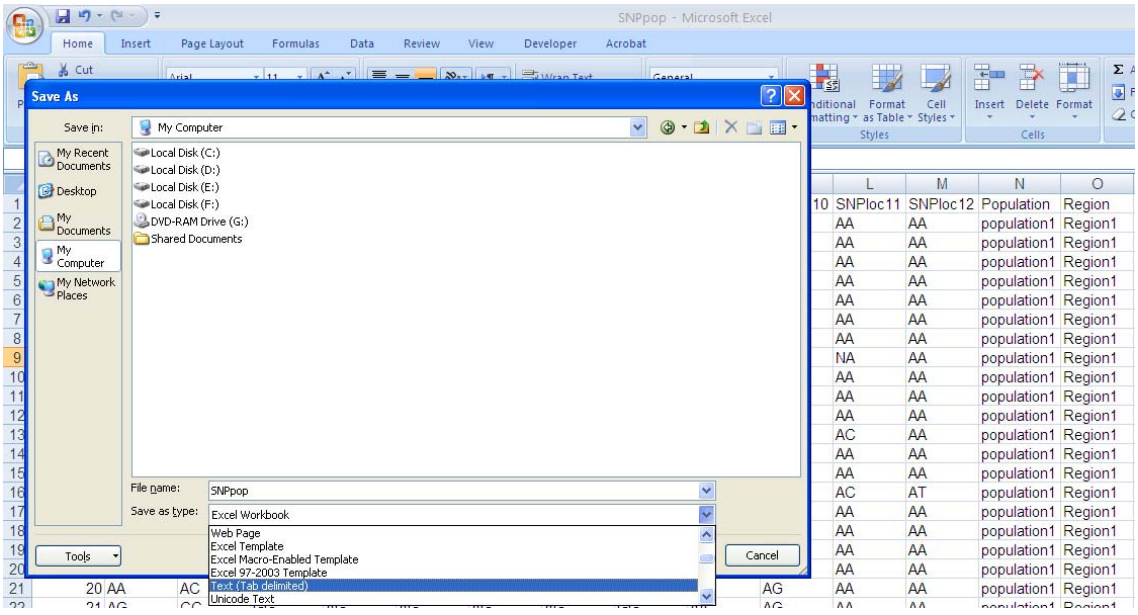
روش‌های مختلفی برای تهیه و تنظیم فایل‌های شخصی از داده‌ها و خوانش آنها در پکیج‌های تجزیه ژنتیکی وجود دارد. همچنین با استفاده از دستورات و توابعی که بدین منظور ایجاد شده‌اند، می‌توان فایل‌هایی که با فرمت سایر نرم‌افزارها تهیه و ذخیره شده است را، خواند. امکانات، محدودیت‌ها و نحوه خوانش فایل‌های ایجاد شده توسط سایر نرم‌افزارها، در بخشی دیگر توضیح داده خواهد. در اینجا ساده‌ترین روشی که برای ایجاد فایل‌های حاوی داده‌های شخصی به منظور خوانش توسط پکیج‌های تجزیه ژنتیکی در R می‌توان بکار برد، شرح داده می‌شود. از آنجا که نحوه تنظیم و خوانش داده‌ها برای نشانگرهای غالب (مانند RAPD و AFLP) کمی متفاوت از نشانگرهای همبارز (مانند SSR و SNP) می‌باشد، روش انجام کار در مورد این دو نوع از نشانگرها در دو بخش جداگانه توضیح داده می‌شود.

همانطور که در شکل ۳-۴ مشخص است، سه جزء اصلی در مجموعه داده‌ی نشانگرها عبارت از شناسه (اسامی) افراد، آلل‌ها به تفکیک لوکوس‌ها و جمعیت‌های انتسابی افراد است. سایر اطلاعات اضافی مانند ناحیه جمع‌آوری افراد و غیره در ستون‌های دیگر درج می‌شود.

بنابراین در گام اول لازم است این اطلاعات به شیوه نشان داده شده در شکل ۳-۴، در یک نرم‌افزار صفحه گسترده مانند Exel وارد شود. دقت کنید که ردیف اول به سرستون‌ها اختصاص دارد که به تفکیک مربوط به اسامی افراد، لوکوس‌ها، جمعیت‌ها و غیره خواهد بود. بنابراین داده‌ها باید از ردیف دوم وارد شوند. پس از تهیه جدول داده‌ها به طریق فوق، فایل را بصورت tab delimited با نام و آدرس دلخواه (مثلاً در درایو :D) ذخیره نمایید (شکل ۳-۵). در اینصورت فایل داده‌ها توسط تابع read.delim قابل خوانش خواهد بود.



شکل ۳-۴- اجزاء اصلی در مجموعه داده‌ی نشانگرهای ژنتیکی

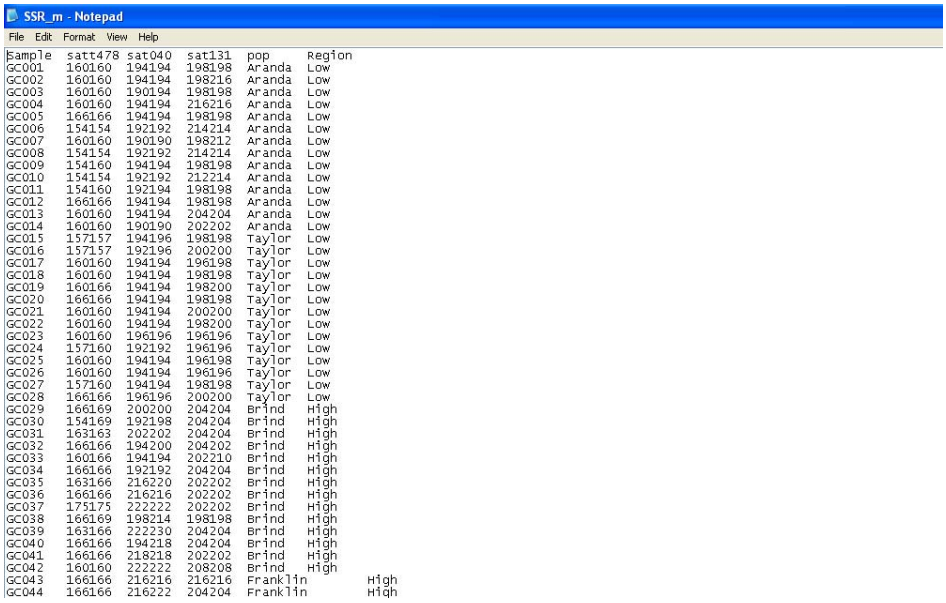


شکل ۳-۵– نحوه ذخیره فایل Exel داده‌های ژنتیکی با فرمت tab delimited

نحوه خوانش فایل داده‌ها برای نشانگرهای همبارز

نشانگرهای میکروستلایت

اگر مراحل تنظیم فایل داده‌ها به درستی انجام گرفته باشد، فایل متنی ذخیره شده باید ساختاری شبیه به شکل ۳-۶ داشته باشد:



شکل ۳-۶- فایل متنی داده‌های نشانگر میکروستلایت برای خوانش توسط پکیج adegenet از نرم‌افزار R

برای تمرین، فایل متنی فوق قبلاً در CD ضمیمه کتاب به نام ssr_m (با پسوند txt.) ذخیره شده است. آن را در درایو :D کپی کنید تا ادامه دستورات قابل اجرا باشد. همانطور که اشاره شد این فایل متنی توسط تابع read.delim قابل خوانش خواهد بود:

```
> library(adegenet)
> a<-read.delim("D:/ssr_m.txt")
> a
```

	Sample	satt478	sat040	sat131	pop	Region
1	GC001	160160	194194	198198	Aranda	Low
2	GC002	160160	194194	198216	Aranda	Low
3	GC003	160160	190194	198198	Aranda	Low
4	GC004	160160	194194	216216	Aranda	Low
5	GC005	166166	194194	198198	Aranda	Low
6	GC006	154154	192192	214214	Aranda	Low

7	GC007	160160	190190	198212	Aranda	Low
8	GC008	154154	192192	214214	Aranda	Low
9	GC009	154160	194194	198198	Aranda	Low
10	GC010	154154	192192	212214	Aranda	Low
11	GC011	154160	192194	198198	Aranda	Low
12	GC012	166166	194194	198198	Aranda	Low
13	GC013	160160	194194	204204	Aranda	Low
14	GC014	160160	190190	202202	Aranda	Low
15	GC015	157157	194196	198198	Taylor	Low
16	GC016	157157	192196	200200	Taylor	Low
17	GC017	160160	194194	196198	Taylor	Low
18	GC018	160160	194194	198198	Taylor	Low
19	GC019	160166	194194	198200	Taylor	Low
20	GC020	166166	194194	198198	Taylor	Low
21	GC021	160160	194194	200200	Taylor	Low
22	GC022	160160	194194	198200	Taylor	Low
23	GC023	160160	196196	196196	Taylor	Low
24	GC024	157160	192192	196196	Taylor	Low
25	GC025	160160	194194	196198	Taylor	Low
26	GC026	160160	194194	196196	Taylor	Low
27	GC027	157160	194194	198198	Taylor	Low
28	GC028	166166	196196	200200	Taylor	Low
29	GC029	166169	200200	204204	Brind	High
30	GC030	154169	192198	204204	Brind	High
31	GC031	163163	202202	204204	Brind	High
32	GC032	166166	194200	204202	Brind	High
33	GC033	160166	194194	202210	Brind	High
34	GC034	166166	192192	204204	Brind	High
35	GC035	163166	216220	202202	Brind	High
36	GC036	166166	216216	202202	Brind	High
37	GC037	175175	222222	202202	Brind	High
38	GC038	166169	198214	198198	Brind	High
39	GC039	163166	222230	204204	Brind	High
40	GC040	166166	194218	204204	Brind	High
41	GC041	166166	218218	202202	Brind	High
42	GC042	160160	222222	208208	Brind	High
43	GC043	166166	216216	216216	Franklin	High
44	GC044	166166	216222	204204	Franklin	High
45	GC045	163166	220220	204204	Franklin	High
46	GC046	166166	216216	216216	Franklin	High
47	GC047	166166	214216	204204	Franklin	High
48	GC048	166169	210212	204204	Franklin	High
49	GC049	166166	212212	204204	Franklin	High
50	GC050	166166	216216	214214	Franklin	High
51	GC051	166166	212212	204214	Franklin	High
52	GC052	154160	216216	204204	Franklin	High
53	GC053	166166	228238	202216	Franklin	High
54	GC054	166166	238238	206206	Franklin	High

55	GC055	166169	230230	202202	Franklin	High
56	GC056	166175	232232	204204	Franklin	High

```
> class(a)
```

```
[1] "data.frame"
```

همانطور که مشاهده می‌شود تا این مرحله، شیء ایجاد شده حاصل از خوانشِ فایل متنی، از نوع قالب داده (data frame) می‌باشد. با مقایسه این شیء با محتویات فایل متنی، مشاهده می‌شود که یک ستون به ابتدای داده‌ها اضافه شده است که در واقع شماره ردیف در قالب داده است. بنابراین برای اینکه بدانیم اولین ستون از نظر R کدام است، دستور زیر را وارد می‌کنیم:

```
> head(a[,1, drop=FALSE])
```

	Sample
1	GC001
2	GC002
3	GC003
4	GC004
5	GC005
6	GC006

براساس نتایج فوق ستون‌های satt478، sat040 و sat131 که مربوط به لوکوس‌ها می‌باشد، به عنوان ستون‌های دوم تا چهارم محسوب خواهند شد. اکنون این ستون‌ها (مربوط به لوکوس‌ها) را در قالب داده‌ای جداگانه ذخیره می‌کنیم:

```
> b<-data.frame(a[,2:4])
> head(b)
```

	satt478	sat040	sat131
1	160160	194194	198198
2	160160	194194	198216
3	160160	190194	198198
4	160160	194194	216216
5	166166	194194	198198
6	154154	192192	214214

اکنون شناسه ردیف‌های قالب داده اخیر را همان اسامی افراد قرار می‌دهیم:

```
> row.names(b) <-a[,1]
> head(b)
```

	satt478	sat040	sat131
GC001	160160	194194	198198
GC002	160160	194194	198216
GC003	160160	190194	198198
GC004	160160	194194	216216
GC005	166166	194194	198198
GC006	154154	192192	214214

اکنون قالب داده فوق، آماده‌ی تبدیل به شیء genind است. این کار با دستور df2genind، قابل انجام است:

```
> c <- df2genind(b, ploidy=2, ncode=3)
```

از آنجا که در فایل داده‌ها برای هر آلل، سه حرف (مثلاً 160) در نظر گرفته شده است، هر لوکوس (دیپلوئید) متشکل که از ۶ حرف می‌باشد (مثلاً 160160) که سه حرف اول آن مربوط به آلل اول و سه حرف دوم آن مربوط به آلل دوم خواهد بود. لذا با قرار دادن عدد ۳ در آرگومان ncode، مشخص می‌کنیم که سه کاراکتر اول برای آلل اول در نظر گرفته شود. بدین ترتیب هر سه کاراکتر برای یک آلل منظور می‌شود، لذا کاراکترهای چهارم تا ششم در مجموع، به عنوان آلل دوم مدنظر قرار خواهند گرفت.

```
> c
/// GENIND OBJECT /////////
 // 56 individuals; 3 loci; 36 alleles; size: 15.5 Kb
 // Basic content
   @tab:  56 x 36 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 7-18)
   @loc.fac: locus factor for the 36 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 2-2)
   @type:  codom
   @call: df2genind(X = b, ncode = 3, ploidy = 2)
 // Optional content
```

- empty -

> tab(c)[1:5,1:5]

	satt478.160	satt478.166	satt478.154	satt478.157	satt478.169
GC001	2	0	0	0	0
GC002	2	0	0	0	0
GC003	2	0	0	0	0
GC004	2	0	0	0	0
GC005	0	2	0	0	0

در این مرحله بایستی جمعیت انتسابی افراد را به شیء genind فوق‌الذکر، اضافه نماییم. داده‌های جمعیت‌ها در ستون پنجم از فایل داده‌ها موجود است که با استفاده از تابع pop، آن را به شیء genind، اضافه می‌نماییم:

> pop(c) <-a[,5]

> c

/// GENIND OBJECT /////////

// 56 individuals; 3 loci; 36 alleles; size: 16.1 Kb

// Basic content

@tab: 56 x 36 matrix of allele counts

@loc.n.all: number of alleles per locus (range: 7-18)

@loc.fac: locus factor for the 36 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: df2genind(X = b, ncode = 3, ploidy = 2)

// Optional content

@pop: population of each individual (group size range: 14-14)

همانطور که در نتایج فوق مشاهده می‌شود یک ردیف به خروجی اضافه شده است (@pop) که نشان‌دهنده‌ی تعریف شدن جمعیت‌های انتسابی افراد است (در مرحله قبل، برای شیء c در سطر آخر empty درج شده بود). سطوح جمعیت‌ها (انواع آن‌ها) با استفاده از تابع (levels(pop() مشخص می‌شود:

> levels(pop(c))

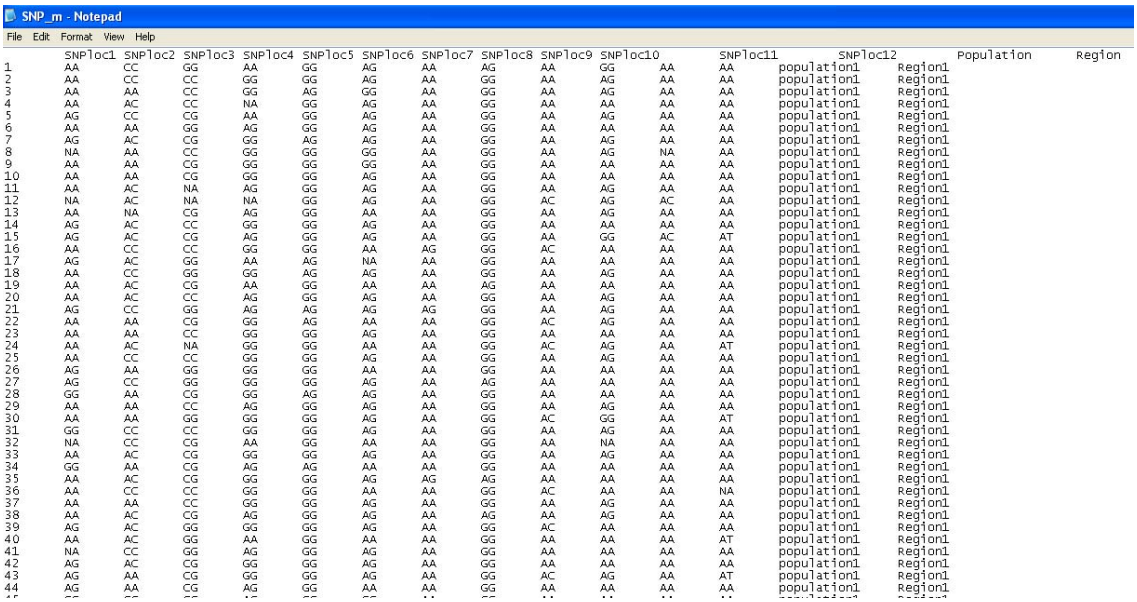
```
[1] "Aranda"   "Taylor"   "Brind"    "Franklin"
```

برای حصول اطمینان از خوانش صحیح آلل‌ها می‌توان از تابع getAlleles استفاده کرد. ولی تابع getAlleles، جزو پکیج pegas است و بر روی اشیاء loci عمل می‌کند. بنابراین لازم است قبل از انجام این کار، با استفاده از تابع genind2loci، شیء genind را به شیء loci تبدیل به نماییم (توجه کنید که باید پکیج pegas را از قبل با دستور install.packages نصب کرده باشید).

```
> library(pegas)
> d<-genind2loci(c)
> getAlleles(d)
$satt478
[1] "154" "169" "157" "160" "166" "163" "175"

$sat040
 [1] "190" "192" "196" "198" "194" "200" "218" "214" "202" "210" "212" "216"
[13] "220" "222" "230" "228" "238" "232"

$sat131
 [1] "196" "198" "200" "212" "216" "202" "210" "204" "206" "208" "214"
```

نشانگرهای SNP

نشانگرهای SNP نیز از نوع همبارز هستند و بنابراین اصول تنظیم فایل و خوانش داده‌های آنها مانند نشانگرهای میکروستلایت که در فوق شرح داده شد (شکل ۳-۷). برای تمرین، فایل snp_m به صورت متنی (با پسوند txt.) در CD همراه کتاب ذخیره شده است. آن را در درایو :D کپی کنید تا ادامه‌ی دستورات، قابل اجرا باشد.



SNP_m - Notepad

File Edit Format View Help

	SNPloc1	SNPloc2	SNPloc3	SNPloc4	SNPloc5	SNPloc6	SNPloc7	SNPloc8	SNPloc9	SNPloc10	SNPloc11	SNPloc12	Population	Region
1	AA	CC	GG	AA	GG	AG	AA	AG	AA	GG	AA	AA	population1	Region1
2	AA	CC	CC	GG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
3	AA	AA	CC	GG	AG	GG	AA	GG	AA	AG	AA	AA	population1	Region1
4	AA	AC	CC	NA	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
5	AG	CC	CG	AA	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
6	AA	AA	GG	AG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
7	AG	AC	CG	GG	AG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
8	NA	AA	CC	GG	GG	GG	AA	GG	AA	AG	NA	AA	population1	Region1
9	AA	AA	CG	GG	GG	GG	AA	GG	AA	AA	AA	AA	population1	Region1
10	AA	AA	CG	GG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
11	AA	AC	NA	AG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
12	NA	AC	NA	NA	GG	AG	AA	GG	AC	AG	AC	AA	population1	Region1
13	AA	NA	CG	AG	GG	AA	AA	GG	AA	AG	AA	AA	population1	Region1
14	AG	AC	CC	GG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
15	AG	AC	CG	AG	GG	AG	AA	GG	AA	GG	AC	AT	population1	Region1
16	AA	CC	CC	GG	GG	AA	AG	GG	AC	AA	AA	AA	population1	Region1
17	AG	AC	GG	AA	AG	NA	AA	GG	AA	AA	AA	AA	population1	Region1
18	AA	CC	GG	GG	AG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
19	AA	AC	CG	AA	GG	AA	AA	AG	AA	AA	AA	AA	population1	Region1
20	AA	AC	CC	AG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
21	AG	CC	GG	AG	AG	AG	AG	GG	AA	AG	AA	AA	population1	Region1
22	AA	AA	CG	GG	AG	AA	AA	GG	AC	AG	AA	AA	population1	Region1
23	AA	AA	CC	GG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
24	AA	AC	NA	GG	GG	AA	AA	GG	AC	AG	AA	AT	population1	Region1
25	AA	CC	CC	GG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
26	AG	AA	GG	GG	GG	AA	AA	GG	AA	AA	AA	AA	population1	Region1
27	AG	CC	GG	GG	GG	AG	AA	AG	AA	AA	AA	AA	population1	Region1
28	GG	AA	CG	GG	AG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
29	AA	AA	CC	AG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
30	AA	AA	GG	GG	GG	AG	AA	GG	AC	GG	AA	AT	population1	Region1
31	GG	CC	CC	GG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
32	NA	CC	CG	AA	GG	AA	AA	GG	AA	NA	AA	AA	population1	Region1
33	AA	AC	CG	GG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
34	GG	AA	CG	AG	AG	AA	AA	GG	AA	AA	AA	AA	population1	Region1
35	AA	AC	CG	GG	GG	AG	AG	AG	AA	AA	AA	AA	population1	Region1
36	AA	CC	CC	GG	GG	AA	AA	GG	AC	AA	AA	NA	population1	Region1
37	AA	AA	CC	GG	GG	AG	AA	GG	AA	AG	AA	AA	population1	Region1
38	AA	AC	CG	AG	GG	AG	AA	AG	AA	AG	AA	AA	population1	Region1
39	AG	AC	GG	GG	GG	AG	AA	GG	AC	AA	AA	AA	population1	Region1
40	AA	AC	GG	AA	GG	AA	AA	GG	AA	AA	AA	AT	population1	Region1
41	NA	CC	GG	AG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
42	AG	AC	CG	GG	GG	AG	AA	GG	AA	AA	AA	AA	population1	Region1
43	AG	AA	CG	GG	GG	AG	AA	GG	AC	AG	AA	AT	population1	Region1
44	AG	AA	CG	AG	GG	AA	AA	GG	AA	AA	AA	AA	population1	Region1

شکل ۳-۷- فایل متنی داده‌های نشانگر SNP برای خوانش توسط پکیج adegenet از نرم‌افزار R

دستوراتی که برای خوانش فایل فوق اجرا می‌کنیم از همان منطق و روال گفته شده در مورد نشانگرهای میکروستلایت تبعیت می‌کند با این تفاوت که در فایل داده‌های SNP برای هر آلل یک حرف (مثلاً A) در نظر گرفته شده است و لذا هر لوکوس (دیپلوئید) متشکل از دو حرف (مثلاً AG) می‌باشد. بنابراین با قرار دادن عدد یک در آرگومان ncode، مشخص می‌کنیم که کاراکتر اول برای آلل اول در نظر گرفته شود و با توجه به اینکه با آرگومان ploidy=2، ارگانیسم مورد بررسی، دیپلوئید تعریف شده است، کاراکتر دوم برای آلل دوم در نظر گرفته خواهد شد.

```
> library(pegas)
> library(adegenet)
> a<-read.delim("D:/snp_m.txt")
> head(a)
```

	X	SNPloc1	SNPloc2	SNPloc3	SNPloc4	SNPloc5	SNPloc6	SNPloc7	SNPloc8	SNPloc9
1	1	AA	CC	GG	AA	GG	AG	AA	AG	AA
2	2	AA	CC	CC	GG	GG	AG	AA	GG	AA
3	3	AA	AA	CC	GG	AG	GG	AA	GG	AA
4	4	AA	AC	CC	<NA>	GG	AG	AA	GG	AA
5	5	AG	CC	CG	AA	GG	AG	AA	GG	AA
6	6	AA	AA	GG	AG	GG	AG	AA	GG	AA

	SNPloc10	SNPloc11	SNPloc12	Population	Region
1	GG	AA	AA	population1	Region1
2	AG	AA	AA	population1	Region1
3	AG	AA	AA	population1	Region1
4	AA	AA	AA	population1	Region1
5	AG	AA	AA	population1	Region1
6	AA	AA	AA	population1	Region1

```
> class(a)
```

```
[1] "data.frame"
```

```
> b<-data.frame(a[,2:13])
```

```
> row.names(b) <-a[,1]
```

```
> c <- df2genind(b, ploidy=2, ncode=1)
```

```
> tab(c)[1:10,1:7]
```

	SNPloc1.A	SNPloc1.G	SNPloc2.C	SNPloc2.A	SNPloc3.G	SNPloc3.C	SNPloc4.A
1	2	0	2	0	2	0	2
2	2	0	2	0	0	2	0
3	2	0	0	2	0	2	0
4	2	0	1	1	0	2	NA
5	1	1	2	0	1	1	2
6	2	0	0	2	2	0	1
7	1	1	1	1	1	1	0
8	NA	NA	0	2	0	2	0
9	2	0	0	2	1	1	0
10	2	0	0	2	1	1	0

```
> pop(c)<-a[,14]
```

```
> levels(pop(c))
```

```
[1] "population1" "population2" "population3" "population4" "population5"
```

```
> d<-genind2loci(c)
```

```
> getAlleles(d)
```

```
$SNPloc1
[1] "A" "G"

$SNPloc2
[1] "A" "C"
```

.....

$SNPloc11

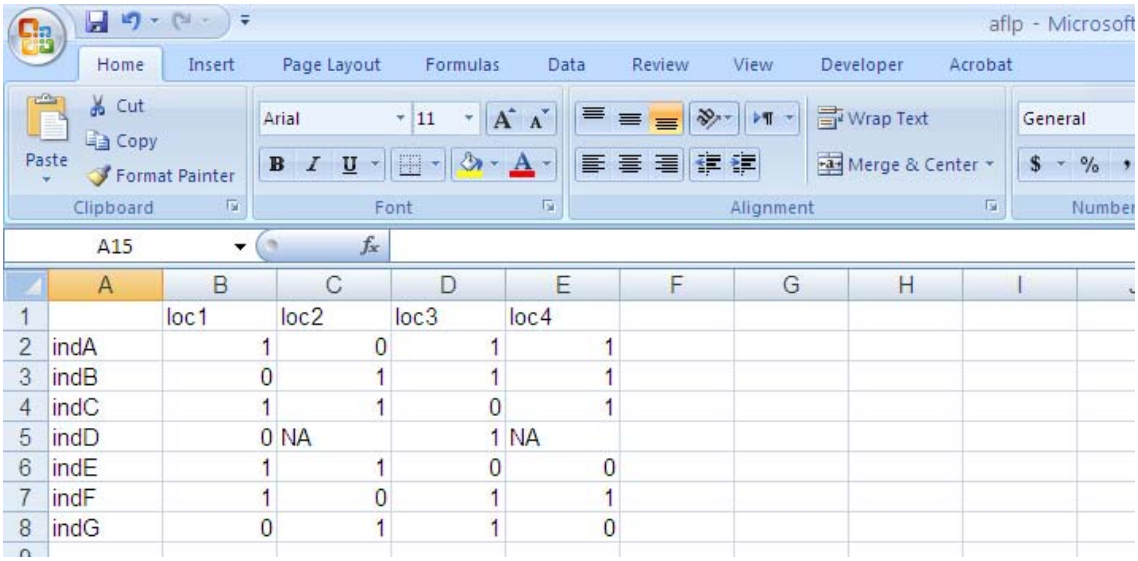
[1] "A" "C"

$SNPloc12

[1] "A" "T"

نحوه تنظیم و خوانش داده‌ها برای نشانگرهای غالب

داده‌های نشانگرهای غالب (مانند RAPD و AFLP) اغلب به صورت باینری (صفر و یک) حاکی از وجود یا عدم وجود باند می‌باشند. لذا تنظیم داده‌ها در فایل Excel مشابه با شکل ۳-۸ خواهد بود:



	A	B	C	D	E	F	G	H	I
1		loc1	loc2	loc3	loc4				
2	indA	1	0	1	1				
3	indB	0	1	1	1				
4	indC	1	1	0	1				
5	indD	0	NA	1	NA				
6	indE	1	1	0	0				
7	indF	1	0	1	1				
8	indG	0	1	1	0				

شکل ۳-۸– نحوه تنظیم داده‌های نشانگرهای غالب در صفحه Excel

تفاوت اصلی داده‌های نشانگرهای غالب با همبارز اینست که در این نشانگرها در هر ستون از لوکوس‌ها، صرفاً یک کاراکتر (صفر یا یک) در هر سلول درج شده است و لذا حتی اگر موجود دیپلوئید باشد، داده‌ها از نوع هاپلوئید خواهد بود و دلیل آن هم خصلت ذاتی این نوع نشانگرها (غالب) است که فرد هتروزیگوت از هموزیگوت قابل تشخیص نیست. لذا این موضوع را در تابع df2genind، با آرگومان‌های ploidy=1 و type="PA" که حاکی از غالب بودن نشانگر است (PA، به حضور یا عدم حضور[۱۲] اشاره دارد) مشخص

[12] Presence/Absence

می‌نماییم. سایر مراحل از جمله افزودن اطلاعات جمعیت به شیء genind و غیره مانند نشانگرهای همبارز است و از همان دستورات استفاده می‌شود و لذا به منظور پرهیز از اطاله کلام، از شرح مجدد آنها خودداری می‌شود. برای تمرین، فایل aflp بصورت متنی در CD همراه کتاب (با پسوند txt.) ذخیره شده است. آن را در درایو :D کپی کنید تا ادامه دستورات قابل اجرا باشد:

```
> library(adegenet)
> aflp_d1<-read.delim("D:/aflp.txt")
> aflp_d1
```

	X	loc1	loc2	loc3	loc4
1	indA	1	0	1	1
2	indB	0	1	1	1
3	indC	1	1	0	1
4	indD	0	NA	1	NA
5	indE	1	1	0	0
6	indF	1	0	1	1
7	indG	0	1	1	0

```
> aflp_d2<-data.frame(aflp_d1[,2:5])
> aflp_d2
```

	loc1	loc2	loc3	loc4
1	1	0	1	1
2	0	1	1	1
3	1	1	0	1
4	0	NA	1	NA
5	1	1	0	0
6	1	0	1	1
7	0	1	1	0

```
> row.names(aflp_d2) <-aflp_d1[,1]
> aflp_d2
```

	loc1	loc2	loc3	loc4
indA	1	0	1	1
indB	0	1	1	1
indC	1	1	0	1
indD	0	NA	1	NA
indE	1	1	0	0
indF	1	0	1	1
indG	0	1	1	0

```
> aflp_d3 <- df2genind(aflp_d2, ploidy=1, type="PA")
> aflp_d3
        /// GENIND OBJECT /////////
         // 7 individuals; 4 loci; 4 alleles; size: 2.3 Kb
         // Basic content
           @tab:  7 x 4 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 4-4)
           @ploidy: ploidy of each individual  (range: 1-1)
           @type:  PA
           @call: df2genind(X = aflp_d2, ploidy = 1, type = "PA")
         // Optional content
           - empty –
```

```
> tab(aflp_d3)
```

	loc1	loc2	loc3	loc4
indA	1	0	1	1
indB	0	1	1	1
indC	1	1	0	1
indD	0	NA	1	NA
indE	1	1	0	0
indF	1	0	1	1
indG	0	1	1	0

از آنجا که شیء حاصله از نوع genind است افزودن اطلاعات جمعیت به آن (انتساب جمعیت برای ژنوتیپ‌ها) مشابه با نشانگرهای همبارز قابل انجام است.

خوانش داده‌های ژنتیکی از سایر نرم‌افزارها

ورود داده‌های ژنتیکی از نرم‌افزارهای STRUCTURE (Pritchard *et al.*, 2000)، GENETIX (Belkhir *et al.*, 1996)، FSTAT (Goudet, 2002) و Genepop (Raymond and Rousset, 1995) که در زمره نرم‌افزارهای رایج ژنتیک جمعیت می‌باشند (Excoffier and Heckel, 2006)، توسط توابع پکیج adegenet میسر است. بدین منظور می‌توان به ترتیب از توابع read.structure، read.genetix، read.fstat و read.genepop، استفاده نمود. این توابع داده‌ها را از فرمت‌های مذکور، به شیء genind تبدیل می‌نمایند که

پس از آن، در قالب پکیج adegenet قابل پردازش خواهد بود. همچنین با استفاده از تابع read.PLINK در پکیج adegenet، می‌توان فایل‌های خروجی نرم‌افزار PLINK (با فرمت raw.) را خوانش و به اشیاء genlight (قابل خوانش توسط پکیج adegenet) تبدیل کرد. بعلاوه، فایل‌های VCF (یا variant calling format) توسط تابع read.vcf در پکیج pegas قابل خوانش می‌باشند.

مثال: خوانش مجموعه داده nancycats که به فرمت STRUCTURE (با پسوند str.) ذخیره شده است و ایجاد شیء genind:

```
> obj1 <- read.structure(system.file("files/nancycats.str", package="adegenet"),
onerowperind= FALSE, n.ind=237, n.loc=9, col.lab=1, col.pop=2, ask=FALSE)
> obj1
```

در دستور فوق، آرگومان onerowperind، یک گزاره منطقی مربوط به گزینه کدنویسی در STRUCTURE است که مشخص می‌کند آیا ژنوتیپ‌ها در یک ردیف (TRUE) یا در دو ردیف (FALSE، پیش‌فرض) کددهی شده‌اند. آرگومان‌های n.ind تعداد افراد و n.loc تعداد نشانگرها در مجموعه داده‌ها را مشخص می‌نمایند که هر دو گزینه بطور پیش‌فرض، NULL می‌باشند. آرگومان‌های col.lab و col.pop، شماره ستون‌هایی هستند که به ترتیب برچسب نام ژنوتیپ‌ها و جمعیت‌ها در آنها درج شده است، درصورت عدم وجود برچسب نام، مقدار آنها صفر خواهد بود. آرگومان ask یک گزاره منطقی است که مشخص می‌کند آیا تابع read.structure، اطلاعات اختیاری (optional) در مورد مجموعه داده‌های مربوطه را از کاربر سوال نماید (TRUE ، پیش‌فرض) و یا اینکه تا حد ممکن سکوت کند (FALSE).

نکته: تابع read.structure فقط برای داده‌های دیپلوئید قابل استفاده است. داده‌های هاپلوئید با فرمت STRUCTURE را می‌توان به راحتی با استفاده از توابع read.table و read.csv خواند و سپس با استفاده از تابع df2genind، به شیء genind تبدیل نمود.

مثال: خوانش مجموعه داده nancycats که به فرمت GENETIX (با پسوند gtx.) ذخیره شده است و ایجاد شیء genind:

```
> obj2 <- read.genetix(system.file("files/nancycats.gtx",package="adegenet"))
> obj2
```

مثال: خوانش مجموعه داده nancycats که به فرمت Fstat (با پسوند dat.) ذخیره شده است و ایجاد شیء genind:

```
>obj3 <- read.fstat(system.file("files/nancycats.dat",package="adegenet"))
> obj3
```

نکته: تابع read.fstat فقط برای داده‌های دیپلوئید قابل استفاده است. داده‌های هاپلوئید را می‌توان پس از حذف سرستون‌ها و خطوط POP، با استفاده از توابع read.table و read.csv خواند و سپس با استفاده از تابع df2genind، به شیء genind تبدیل نمود.

مثال: خوانش مجموعه داده nancycats که به فرمت Genepop (با پسوند gen.) ذخیره شده است و ایجاد شیء genind:

```
> obj4 <- read.genepop(system.file("files/nancycats.gen",package="adegenet"))
> obj4
```

نکته: تابع read.genepop فقط برای داده‌های دیپلوئید قابل استفاده است. داده‌های هاپلوئید با فرمت Genepop را می‌توان پس از حذف سرستون‌ها و خطوط POP، با استفاده از توابع read.table و read.csv خواند و سپس با استفاده از تابع df2genind، به شیء genind تبدیل نمود.

برای جزئیات بیشتر در مورد این توابع و آرگومان‌های مربوطه به فایل راهنمای پکیج adgenet موجود در آدرس https://cran.r-project.org/web/packages/adegenet/adegenet.pdf مراجعه نمایید.

همچنین برای ورود داده‌ها از هر یک از فرمت‌های فوق، می‌توان از تابع کلی import2genind استفاده نمود. این تابع، پسوند درج شده در آرگومان را شناسایی می‌کند و به دنبال تابع مناسب برای تبدیل مجموع داده مربوطه به شیء genind، می‌گردد. در حال حاضر چهار فرمت توسط این تابع قابل شناسایی و تبدیل می‌باشند که عبارتند از فایل‌های GENETIX، (با پسوند gtx.)، فایل‌های Genepop، (با پسوند gen.)، فایل‌های Fstat، (با پسوند dat.) و فایل‌های STRUCTURE، (با پسوند str. یا stru.)

مثال: خوانش مجموعه داده nancycats که به فرمت‌های مختلف ذخیره شده است و ایجاد شیء genind:

```
> import2genind(system.file("files/nancycats.gtx", package="adegenet"))
> import2genind(system.file("files/nancycats.dat", package="adegenet"))
> import2genind(system.file("files/nancycats.gen", package="adegenet"))
> import2genind(system.file("files/nancycats.str", package="adegenet"),
onerowperind=FALSE, n.ind=237, n.loc=9, col.lab=1, col.pop=2, ask=FALSE)
```

نکته مهمی که باید در این رابطه به خاطر داشت اینست که نباید انتظار داشت که یک مجموعه داده در فرمت‌های مختلف، به اشیاء genind مشابهی منجر شود. مثلاً تبدیل از فرمت GENETIX به Fstat ممکن است سبب تغییر ترتیب ژنوتیپ‌ها شود. فرمت GENETIX اسم افراد را ذخیره می‌کند ولی Fstat این کار را انجام نمی‌دهد. Genepop نام یک نمونه را از نام آخرین ژنوتیپ انتخاب می‌کند و غیره.

علاوه بر امکانِ ورود داده‌های ژنتیکی از نرم‌افزارهای STRUCTURE، GENETIX، FSTAT و Genepop که در فوق توضیح داده شد، ورود داده‌های GenAlEx (Peakall and Smouse, 2006) توسط تابع read.genalex در پکیج poppr میسر است.

مثال: خوانش مجموعه داده‌های rootrot که به فرمت GenAlEx (با پسوند csv.) ذخیره شده و ایجاد شیء genind:

```
> library(poppr)

> obj5 <- read.genalex(system.file("files/rootrot.csv", package="poppr"), genclone = FALSE)
> obj5
        /// GENIND OBJECT /////////

         // 187 individuals; 56 loci; 56 alleles; size: 55.7 Kb

         // Basic content
           @tab:  187 x 56 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 56-56)
           @ploidy: ploidy of each individual  (range: 2-2)
           @type:  PA
           @call: read.genalex(genalex = "rootrot.csv", genclone = FALSE)

         // Optional content
           @pop: population of each individual (group size range: 5-17)
           @strata: a data frame with 1 columns ( Pop )
```

توجه داشته باشید که درصورت انتخاب گزینه TRUE برای آرگومان genclone، خروجی تابع read.genalex به صورت اشیاء genclone ذخیره خواهد شد. همچنین با استفاده از آرگومان‌های geo=TRUE و region=TRUE، خوانش فایل‌های GenAlEx که به ترتیب دارای مختصات جغرافیایی یا اطلاعات منطقه‌ای هستند، میسر است.

تبدیل اشیاء در پکیج‌های تجزیه ژنتیکی

پکیج‌های تحت R که به حوزه مشابهی می‌پردازند معمولاً دارای توابعی هستند که اشیاء ذیل آنها را به فرمت‌های قابل خوانش توسط پکیج‌های دیگر و برعکس تبدیل می‌کنند. پکیج adegenet، دارای توابع مفیدی برای تبدیل اشیاء genind به اشیاء قابل خوانش توسط سایر پکیج‌ها می‌باشد. از تابع genind2df برای تبدیل اشیاء genind به قالب داده و از تابع df2genind برای عکس این عمل، می‌توان استفاده نمود. بدین ترتیب می‌توان ژنوتیپ‌هایی را که بصورت رشته‌های کاراکتری، کدنویسی و در قالب data.frame

(ژنوتیپ‌ها در ردیف و نشانگرها در ستون) ذخیره شده‌اند را، خوانش نمود. برای این منظور از تابع df2genind در پکیج adegenet استفاده می‌شود. این تابع برای هر سطحی از پلوئیدی قابل اجرا می‌باشد. مثال: ابتدا یک فایل قالب داده (به نام df) تشکیل می‌دهیم و سپس آن را به شیء genind تبدیل می‌نماییم:

```
> library(adegenet)

> df <- data.frame(locusA=c("11","11","12","32"), locusB=c(NA,"34","55","15"),
locusC=c("22","22","21","22"))

> row.names(df) <- .genlab("genotype",4)

> df
```

	locusA	locusB	locusC
genotype1	11	<NA>	22
genotype2	11	34	22
genotype3	12	55	21
genotype4	32	15	22

```
> obj6 <- df2genind(df, ploidy=2, ncode=1)

> obj6

        /// GENIND OBJECT /////////

         // 4 individuals; 3 loci; 9 alleles; size: 3.5 Kb

         // Basic content

           @tab:  4 x 9 matrix of allele counts

           @loc.n.all: number of alleles per locus (range: 2-4)

           @loc.fac: locus factor for the 9 columns of @tab

           @all.names: list of allele names for each locus

           @ploidy: ploidy of each individual  (range: 2-2)

           @type:  codom

           @call: df2genind(X = df, ncode = 1, ploidy = 2)

         // Optional content

           - empty -

> tab(obj6)
```

	locusA.1	locusA.2	locusA.3	locusB.3	locusB.4	locusB.5	locusB.1	locusC.2
1	2	0	0	NA	NA	NA	NA	2

2	2	0	0	1	1	0	0	2
3	1	1	0	0	0	2	0	1
4	0	1	1	0	0	1	1	2

	locusC.1
1	0
2	0
3	1
4	0

```
> obj7 <- genind2df(obj6)
> obj7
```

	locusA	locusB	locusC
1	11	<NA>	22
2	11	34	22
3	12	55	21
4	23	51	22

از توابع genind2loci و as.loci در پکیج pegas، برای تبدیل اشیاء genind به فرمت قابل خوانش توسط پکیج pegas (با فرمت loci) و از تابع loci2genind در پکیج pegas برای تبدیل اشیاء loci به فرمت قابل خوانش توسط پکیج adegenet (با فرمت genind)، می‌توان استفاده نمود. مثال:

```
> library(adegenet)
> library(pegas)
> x<-as.loci(nancycats)
> x
        Allelic data frame: 237 individuals
                    9 loci
                    1 additional variable
> class(x)
        [1] "loci"      "data.frame"
> y<-loci2genind(x)
> class(y)
        [1] "genind"
        attr(,"package")
```

```
[1] "adegenet"
> z<-genind2loci(y)
> class(z)
[1] "loci"      "data.frame"
```

از تابع as.loci در پکیج pegas، همچنین می‌توان برای تبدیل اشیاء قالب داده به فرمت قابل خوانش توسط پکیج pegas (اشیاء loci) استفاده نمود:

```
> a<-genind2df(nancycats)
> class(a)
[1] "data.frame"
> b<-as.loci(a)
> class(b)
[1] "loci"      "data.frame"
```

پرداختن به مقادیر گمشده

شناسایی مقادیر گمشده

وجود مقادیر نامعلوم در مجموعه داده‌ها که به عنوان مقادیر گمشده معروف می‌باشند، امری معمول است. این موضوع گاهی تجزیه ژنتیکی را دچار اشکال می‌سازد چراکه برخی توابع درصورت وجود مقادیر گمشده عمل نخواهند نمود. مثال:

```
> library(adegenet)
> dudi.pca(nancycats)
Error in dudi.pca(nancycats) : na entries in table
```

لذا به منظور تشخیص مقادیر گمشده و جایگزینی آنها، توابعی ایجاد شده‌اند. همانطور که قبلاً نیز اشاره شد، با استفاده از تابع summary می‌توان به وجود مقادیر گمشده و فراوانی آنها پی برد. به عنوان مثال در مجموعه داده nancycats میزان ۲/۳۴ درصد، داده گمشده وجود دارد:

```
> library(adegenet)
> data(nancycats)
```

```
> summary(nancycats)
    // Number of individuals: 237
    // Group sizes: 10 22 12 23 15 11 14 10 9 11 20 14 13 17 11 12 13
    // Number of alleles per locus: 16 11 10 9 12 8 12 12 18
    // Number of alleles per group: 36 53 50 67 48 56 42 54 43 46 70 52 44 61 42 40 35
    // Percentage of missing data: 2.34 %
    // Observed heterozygosity: 0.67 0.67 0.68 0.71 0.63 0.57 0.65 0.62 0.45
    // Expected heterozygosity: 0.87 0.79 0.8 0.76 0.87 0.69 0.82 0.76 0.61
```

از تابع عمومی is.na می‌توان برای شناسایی مقادیر گمشده استفاده کرد. خروجی این تابع درصورت وجود مقدار گمشده، TRUE و در غیراینصورت، FALSE خواهد بود:

```
> is.na(c(1, NA))
    [1] FALSE  TRUE
> is.na(c(3,2, NA,7))
    [1] FALSE FALSE  TRUE FALSE
```

با استفاده از تابع عمومی anyNA نیز می‌توان بطور کلی وجود یا عدم وجود داده گمشده را بررسی نمود:

```
> anyNA(c(1, 2,6,8))
    [1] FALSE
> anyNA(c(3,2, NA,7))
    [1] TRUE
```

برای استفاده از توابع فوق بر روی اشیاء genind، لازم است آنها را در ترکیب با تابع tab بکار ببریم:

```
> tab(nancycats)[1:5,1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA
N217	0	0	0	0	0
N218	0	0	0	0	0
N219	0	0	0	0	0

```
> is.na(tab(nancycats)[1:5,1:5])
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127

N215	TRUE	TRUE	TRUE	TRUE	TRUE
N216	TRUE	TRUE	TRUE	TRUE	TRUE
N217	FALSE	FALSE	FALSE	FALSE	FALSE
N218	FALSE	FALSE	FALSE	FALSE	FALSE
N219	FALSE	FALSE	FALSE	FALSE	FALSE

با استفاده از تابع which می‌توان افرادِ واجد مقادیر گمشده را متمایز نمود:

```
> which(is.na(tab(nancycats)[1:5,1:5]), TRUE)
```

	row	col
N215	1	1
N216	2	1
N215	1	2
N216	2	2
N215	1	3
N216	2	3
N215	1	4
N216	2	4
N215	1	5
N216	2	5

با استفاده از تابع propTyped می‌توان نسبت داده‌های موجود (غیرگمشده) در اشیاء genind و genpop را محاسبه نمود:

برای اشیاء genind:

```
propTyped(x, by=c("ind","loc","both"))
```

برای اشیاء genpop:

```
propTyped(x, by=c("pop","loc","both"))
```

آرگومان by مشخص می‌کند که نسبت داده‌های موجود (غیرگمشده) برای افراد (ind)، لوکوس‌ها (loc)، جمعیت‌ها (pop) یا هر دو (در مورد اشیاء genind ، افراد و لوکوس‌ها و در مورد اشیاء genpop، جمعیت‌ها و لوکوس‌ها) محاسبه شود.

مثال: در مجموعه داده nancycats نسبت داده‌های موجود (غیرگمشده) را برحسب افراد و لوکوس‌ها را محاسبه می‌کنیم:

```
> library(adegenet)
> data(nancycats)
```

```
> propTyped(nancycats,by="ind")
```

N215	N216	N217	N218	N219	N220	N221	N222
0.8888889	0.8888889	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
N223	N224	N7	N141	N142	N143	N144	N145
1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
….	….	….	….	….	….	….	….
N282	N283	N288	N291	N292	N293	N294	N295
0.8888889	0.8888889	0.8888889	0.7777778	0.7777778	0.7777778	0.7777778	0.7777778
	N296	N297	N281	N289	N290		
0.7777778	0.7777778	0.8888889	0.8888889	0.7777778			

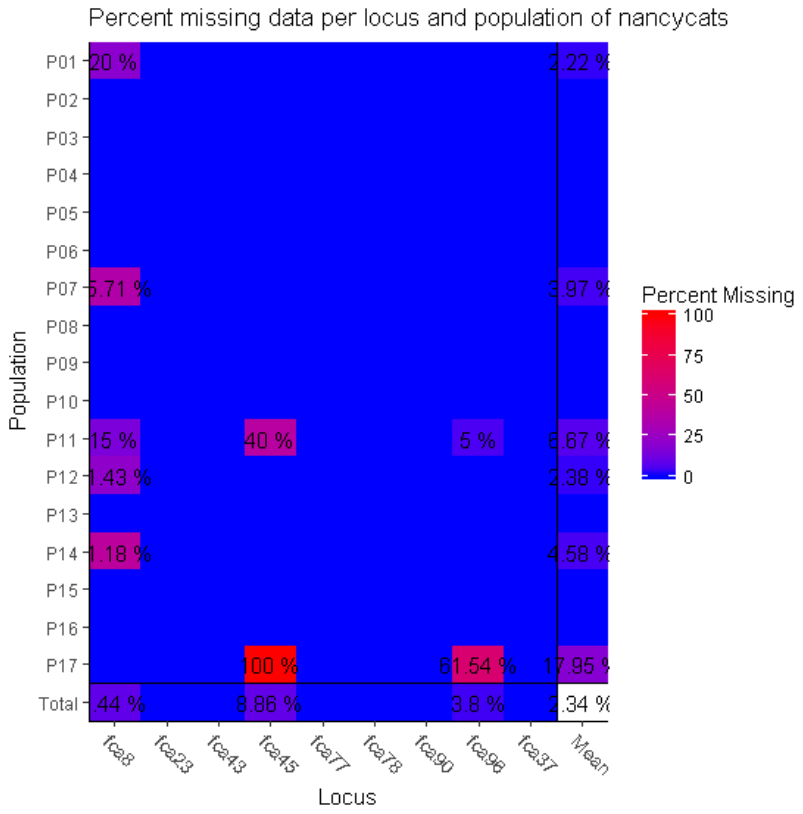
```
> propTyped(nancycats,by="loc")
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
0.915612	1.000000	1.000000	0.911392	1.000000	1.000000	1.000000	0.962025	1.000000

همچنین تابع info_table در پکیج poppr، تابع مفیدی برای شناسایی مقادیر گمشده و ارئه تصویری از وضعیت آنها در جمعیت‌ها و لوکوس‌های مختلف می‌باشد (شکل ۳-۹):

```
> library(adegenet)
> library(poppr)
> data(nancycats)
> nancy.miss <- info_table(nancycats, plot = TRUE)
>  nancy.miss
```

	Locus									
Population	fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37	Mean
P01	0.200	.	.	.	.	.	.	.	.	0.022
P02	.	.	.	.	.	.	.	.	.	.
P03	.	.	.	.	.	.	.	.	.	.
……..										
P14	0.412	.	.	.	.	.	.	.	.	0.046
P15	.	.	.	.	.	.	.	.	.	.
P16	.	.	.	.	.	.	.	.	.	.
P17	.	.	.	1.000	.	.	.	0.615	.	0.179
Total	0.084	.	.	0.089	.	.	.	0.038	.	0.023



شکل ۳-۹– نمایش نسبت مقادیر گمشده به تفکیک لوکوس‌ها و جمعیت‌ها در مجموعه داده nancycats

جایگزینی مقادیر گمشده

از تابع tab می‌توان برای جایگزینی مقادیر گمشده استفاده کرد:

tab(obj, NA.method="mean")

در دستور فوق obj، یک شیء genind است و NA.method نحوه جایگزینی بجای مقدار گمشده را مشخص می‌کند که می‌تواند میانگین ("mean")، یا مقدار صفر ("zero") باشد. به عنوان مثال برای جایگزین کردن مقادیر گمشده در داده‌های nancycats با میانگینِ تعداد آلل‌ها، از گزینه NA.method="mean" و برای جایگزین کردن مقادیر گمشده با صفر از گزینه NA.method="zero" استفاده می‌شود:

```
> library(adegenet)
> tab(nancycats)[1:5,1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	NA	NA	NA	NA	NA
N216	NA	NA	NA	NA	NA
N217	0	0	0	0	0

N218	0	0	0	0	0
N219	0	0	0	0	0

```
> tab(nancycats, NA.method="mean")[1:5,1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	0.004608	0.004608	0.02765	0.133641	0.004608
N216	0.004608	0.004608	0.02765	0.133641	0.004608
N217	0	0	0	0	0
N218	0	0	0	0	0
N219	0	0	0	0	0

```
> tab(nancycats, NA.method="zero")[1:5,1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	0	0	0	0	0
N216	0	0	0	0	0
N217	0	0	0	0	0
N218	0	0	0	0	0
N219	0	0	0	0	0

عمل جایگزینی مقادیر گمشده را هنگام استاندارد کردن داده‌ها نیز می‌توان انجام داد. از تابع scaleGen برای استاندارد کردن داده‌ها استفاده می‌شود:

```
> x<-scaleGen(nancycats, NA.method ="mean")
> head(x)[1:5, 1:5]
```

	fca8.117	fca8.119	fca8.121	fca8.123	fca8.127
N215	0	0	0	0	0
N216	0	0	0	0	0
N217	-0.07096	-0.07096	-0.17586	-0.3942	-0.07096
N218	-0.07096	-0.07096	-0.17586	-0.3942	-0.07096
N219	-0.07096	-0.07096	-0.17586	-0.3942	-0.07096

در نتایج فوق مشاهده می‌شود که مقادیر شمارش (تعداد) آلل‌ها، با مقادیر استاندارد شده، جایگزین شده است و بجای مقادیر گمشده عدد صفر درج گردیده است. دلیل جایگزین شدن مقادیر گمشده با مقدار صفر اینست که با استاندارد کردن داده‌ها، میانگین آنها صفر و واریانس یک خواهد بود، بنابراین آرگومان NA.method ="mean" در دستور فوق، مقادیر گمشده را با میانگین (که در اینجا همان صفر است) جایگزین خواهد کرد.

علاوه بر تنظیمات دستی فوق ذکر، تابع missingno در پکیج poppr، بطور اختصاصی برای پرداختن به مقادیر گمشده ایجاد شده است:

missingno(x, type = "loci", cutoff = 0.05)

با انتخاب گزینه loci برای آرگومان type (پیش‌فرض)، تمام لوکوس‌های حاوی داده گمشده از مجموعه داده‌ها حذف می‌شوند. گزینه genotype برای این آرگومان، سبب حذف تمام ژنوتیپ‌های حاوی داده گمشده می‌شود. با انتخاب گزینه mean و zero ، مقادیر گمشده به ترتیب با میانگین تعداد آلل‌ها یا مقدار صفر در کل مجموعه داده، جایگزین می‌شوند. با گزینه ignore داده‌های گمشده، جایگزین یا حذف نمی‌شوند. همچنین با تعیین مقداری عددی بین صفر و یک برای آرگومان cutoff، می‌توان آستانه‌ای مجاز برای وجود داده گمشده برای ژنوتیپ‌ها یا لوکوس‌ها تعیین کرد. این آستانه مجاز بطور پیش‌فرض ۵ درصد (cutoff = 0.05) است.

```
> library(adegenet)
> library(poppr)
> data(nancycats)
> nancy.locina <- missingno(nancycats, type = "loci")
        Found 617 missing values.
        2 loci contained missing values greater than 5%
        Removing 2 loci: fca8, fca45
> nancy.locina
        /// GENIND OBJECT /////////
         // 237 individuals; 7 loci; 83 alleles; size: 99.1 Kb
         // Basic content
           @tab:  237 x 83 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 8-18)
           @loc.fac: locus factor for the 83 columns of @tab
           @all.names: list of allele names for each locus
           @ploidy: ploidy of each individual  (range: 2-2)
           @type:  codom
           @call: .local(x = x, i = i, j = j, drop = drop)
```

// Optional content

@pop: population of each individual (group size range: 9-23)

@other: a list containing: xy

```
> locNames(nancy.locina)
```

[1] "fca23" "fca43" "fca77" "fca78" "fca90" "fca96" "fca37"

همانطور که در نتایج فوق مشخص است در شیء nancy.locina، دو لوکوس fca8 و fca45 از مجموع ۹ لوکوس موجود در مجموعه داده nancycats، حذف شده است.

```
> nancy.genona <- missingno(nancycats, type = "geno")
```

Found 617 missing values.

38 genotypes contained missing values greater than 5%

Removing 38 genotypes: N215, N216, N188, N189, N190, N191, N192, N298, N299, N300, N301, N302, N303, N304, N310, N195, N197, N198, N199, N200, N201, N206, N182, N184, N186, N282, N283, N288, N291, N292, N293, N294, N295, N296, N297, N281, N289, N290

```
> nancy.0 <- missingno(nancycats, type = "0")
```

Replaced 617 missing values.

```
> nancy.mean <- missingno(nancycats, type = "mean")
```

Replaced 617 missing values.

فصل چهارم- آمار توصیفی داده‌های ژنتیکی و آزمون‌های پایه

مشاهده اندازه جمعیت‌ها و رسم نمودار ستونی

همانطور که قبلاً نیز اشاره شد، جمعیت‌های انتسابی افراد را می‌توان با تابع pop مشاهده نمود. به عنوان مثال در مجموعه داده nancycats جمعیت‌هایی که فرد اول تا فرد شماره ۲۳۷ (آخرین فرد) به آنها منتسب می‌باشند، به شرح زیر است:

```
> library(adegenet)
> data(nancycats)
> pop(nancycats)
    [1] P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P02 P02 P02 P02 P02 P02 P02 P02
   [19] P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P03 P03 P03 P03
   ……..
  [217] P16 P12 P12 P12 P12 P12 P12 P12 P17 P17 P17 P17 P17 P17 P17 P17 P17 P17
  [235] P17 P17 P17
  17 Levels: P01 P02 P03 P04 P05 P06 P07 P08 P09 P10 P11 P12 P13 P14 P15 ... P17
```

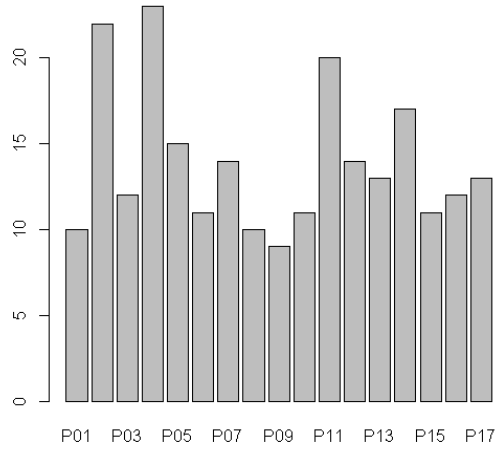
تعداد افراد به تفکیک جمعیت‌ها (اندازه جمعیت‌ها) را می‌توان با تابع table شمارش نمود:

```
> table(pop(nancycats))
```

P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	P16	P17
10	22	12	23	15	11	14	10	9	11	20	14	13	17	11	12	13

با استفاده از تابع barplot می‌توان نمودار ستونی مربوط به تعداد افراد تشکیل‌دهنده جمعیت‌ها را ترسیم نمود (شکل ۴-۱):

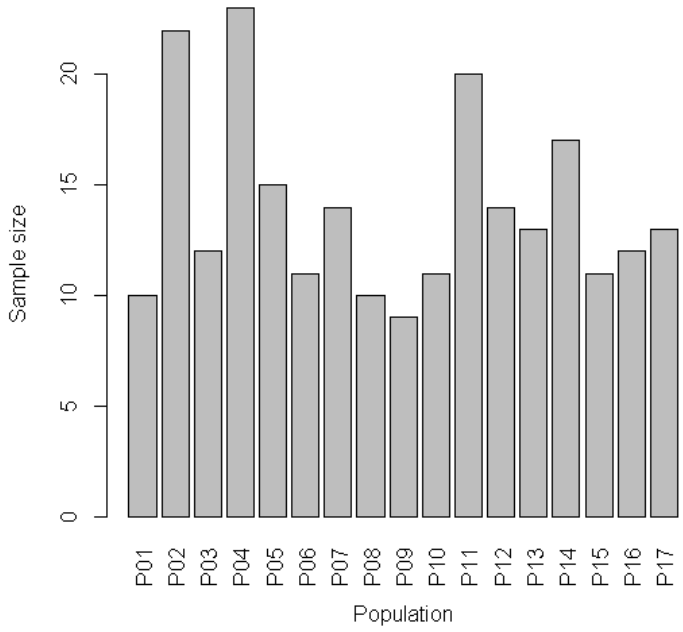
```
> barplot(table(pop(nancycats)))
```



شکل ۴-۱- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats

همانطور که مشاهده می‌شود به دلیل محدودیت فضای پایین محور افقی نمودار، اسامی جمعیت‌ها بطور کامل درج نشده است. با استفاده از آرگومان las=3 می‌توان اسامی جمعیت‌ها را در جهت عمودی درج نمود. همچنین با استفاده از آرگومان‌های xlab و ylab، می‌توان برای محورهای افقی و عمودی عنوان (برچسب) تعیین کرد (شکل ۴-۲):

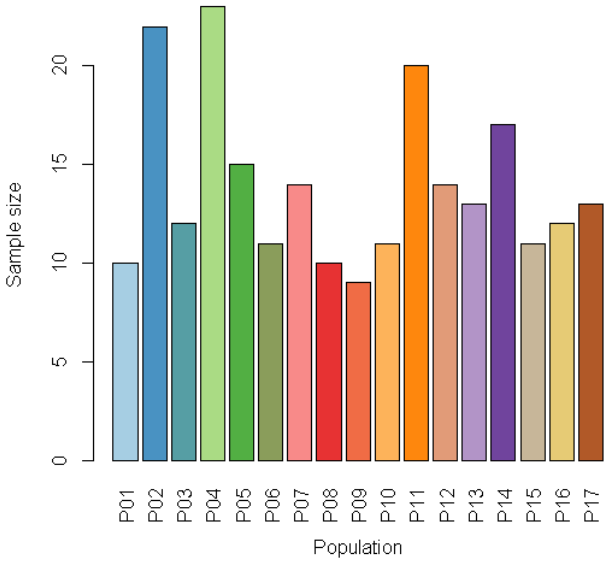
```
> barplot(table(pop(nancycats)), las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۲- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با انجام تنظیمات در آرگومان‌های تابع barplot

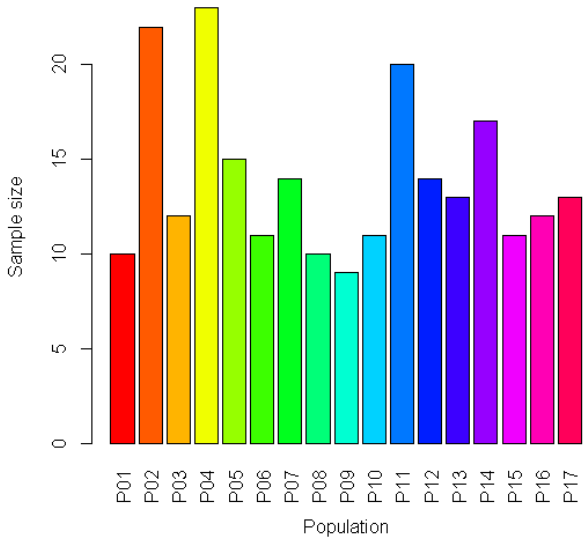
برای رنگ‌آمیزی ستون‌ها در نمودار، به روش‌های مختلف می‌توان عمل کرد که در اینجا سه روش متفاوت که رنگ‌های متنوعی ایجاد می‌نماید، توضیح داده می‌شود. به طور کلی برای تعیین رنگ از آرگومان col استفاده می‌شود. سه تابع مختلف برای این گزینه می‌توان استفاده نمود که عبارت از funky، rainbow و colorRampPalette می‌باشند. تابع funky به طور خودکار رنگ‌های متنوع ایجاد می‌کند (شکل ۴-۳) و تابع rainbow از رنگ‌های رنگین کمانی استفاده می‌کند (شکل ۴-۴). ورودی این دو تابع تعداد ستون‌های نمودار یا همان تعداد کل جمیعت‌ها (۱۷) خواهد بود.

```
> barplot(table(pop(nancycats)), col=funky(17), las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۳- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با بکار بردن تابع funky در آرگومان col از تابع barplot

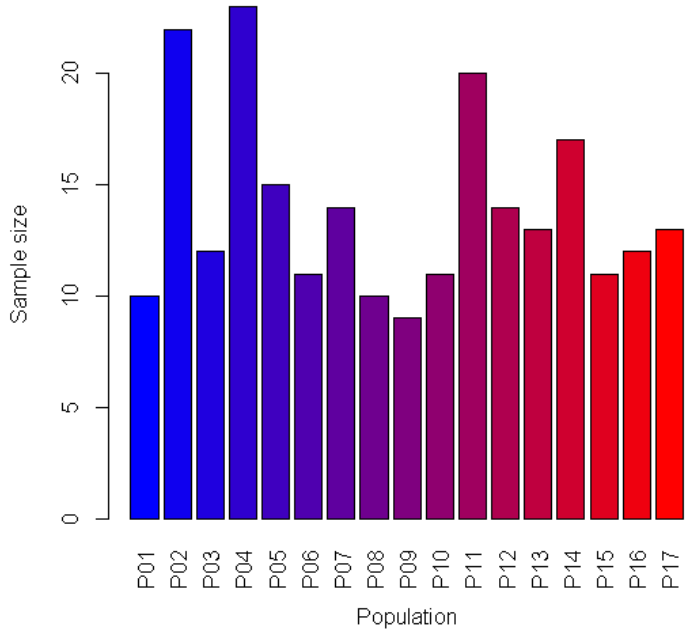
```
> barplot(table(pop(nancycats)), col=rainbow(17), las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۴- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با بکار بردن تابع rainbow در آرگومان col از تابع barplot

با استفاده از تابع colorRampPalette می‌توان طیفی از رنگ بین دو یا چند نوع رنگی که کاربر مشخص می‌کند ایجاد کرد، مثلاً می‌توان طیفی از رنگ آبی به سمت قرمز یا غیره ایجاد کرد (شکل ۴-۵):

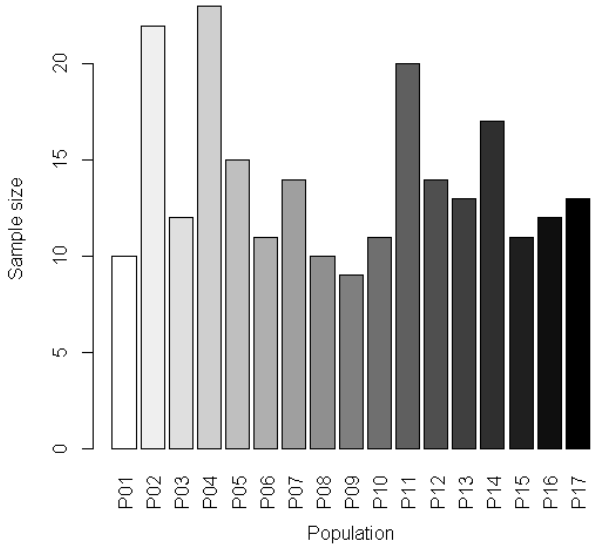
```
> barplot(table(pop(nancycats)), col= colorRampPalette(c("blue", "red"))( 17 ) , las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۵- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با تعیین طیف رنگ آبی تا قرمز برای تابع colorRampPalette در آرگومان col از تابع barplot

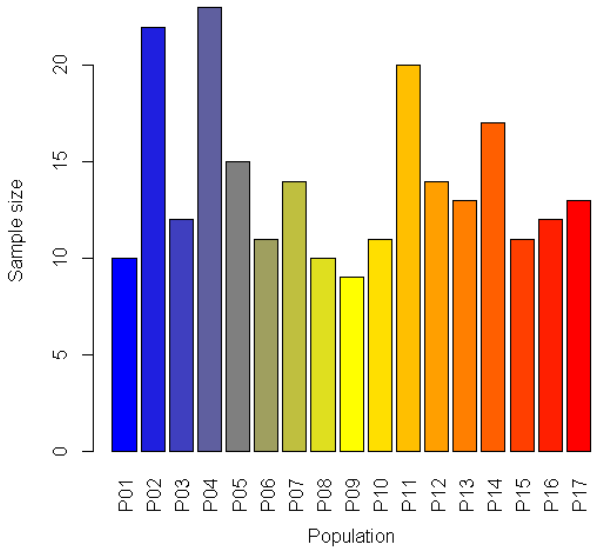
آرگومان‌های تابع colorRampPalette برای ایجاد طیفی از سفید تا سیاه (شکل ۴-۶) و آبی-زرد-قرمز (شکل ۴-۷) بصورت زیر قابل تنظیم می‌باشد:

```
> barplot(table(pop(nancycats)), col= colorRampPalette(c("white", "black"))( 17 ) , las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۶- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با تعیین طیف رنگ سفید تا سیاه برای تابع colorRampPalette در آرگومان col از تابع barplot

```
> barplot(table(pop(nancycats)), col= colorRampPalette(c("blue", "yellow","red"))( 17 ) , las=3, xlab="Population", ylab="Sample size")
```



شکل ۴-۷- نمودار ستونی اندازه جمعیت‌ها در مجموعه داده nancycats با تعیین طیف رنگ آبی-زرد- قرمز برای تابع colorRampPalette در آرگومان col از تابع barplot

بررسی وجود پلی‌مورفیسم

با استفاده از تابع isPoly می‌توان پلی‌مورفیک بودن یک لوکوس یا آلل را بررسی نمود:

isPoly(x, by=c("locus","allele"), thres=1/100)

x شیء genind یا genpop است. با استفاده از آرگومان by بررسی پلی‌مورفیسم برای لوکوس (locus) یا آلل (allele) را تعیین می‌کنیم. آرگومان thres، حداقل فراوانی آللی برای دلالت بر پلی‌مورفیسم را نشان می‌دهد که به عنوان پیش فرض دارای مقدار ۰/۰۱ است. خروجی این تابع وکتوری منطقی است که مقدار TRUE دلالت بر وجود پلی‌مورفیسم دارد و برعکس. مثال:

```
> library(adegenet)
> data(nancycats)
> isPoly(nancycats,by="loc", thres=0.1)
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

```
> isPoly(nancycats,by="allele", thres=0.1)
```

fca8.117	fca8.119	fca8.121	fca8.123	fca8.127	fca8.129	fca8.131	fca8.133
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
fca8.135	fca8.137	fca8.139	fca8.141	fca8.143	fca8.145	fca8.147	fca8.149
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

.....

می‌توان وجود پلی‌مورفیسم را در بخشی از جمعیت (بجای کل آن) نیز بررسی نمود. در مثال زیر، وجود پلی‌مورفیسم فقط در سه فرد اول بررسی شده است و همانطور که مشاهده می‌شود، برخلاف کل جمعیت که تمام لوکوس‌ها، پلی‌مورفیک بودند، در اینجا لوکوس fca96، پلی‌مورفیسم نشان نداده است:

```
> isPoly(nancycats[1:3],by="loc", thres=0.1)
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE

برای مشاهده وضعیت آلل‌ها در دادهای اصلی می‌توان از تبدیل genind به data frame استفاده نمود. در اینحالت عدم وجود پلی‌موفیسم در لوکوس fca96، مشهود است:

```
> genind2df(nancycats[1:3])
```

	pop	fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
N215	P01	<NA>	136146	139139	116120	156156	142148	199199	113113	208208
N216	P01	<NA>	146146	139145	120126	156156	142148	185199	113113	208208
N217	P01	135143	136146	141141	116116	152156	142142	197197	113113	210210

محاسبه فراوانی‌های ژنوتیپی و آللی

برای مشاهده انواع ژنوتیپ‌ها و آلل‌ها در لوکوس‌های مختلف و به دست آوردن مقادیر فراوانی‌های ژنوتیپی و آللی، می‌توان از توابع موجود در پکیج pegas استفاده کرد. برای کار با پکیج pegas لازم است شیء genind به شیء loci تبدیل شود، که این کار با تابع genind2loci قابل انجام است. به عنوان مثال برای مشاهده فراوانی‌های ژنوتیپی و آللی در مجموعه داده nancycats به شرح زیر عمل می‌نماییم:

```
> library(adegenet)
> library(pegas)
> data(nancycats)
> l.cats<-genind2loci(nancycats)
> l.cats
        Allelic data frame: 237 individuals
                    9 loci
                    1 additional variable
> class(l.cats)
        [1] "loci"      "data.frame"
```

همانطور که مشاهده می‌شود شیء جدیداً تشکیل شده (l.cats) از نوع loci است. اجزای تشکیل دهنده این شیء را می‌توان با دستور str مشاهده نمود:

```
> str(l.cats)
Classes ‘loci’ and 'data.frame':	237 obs. of  10 variables:
 $ population: Factor w/ 17 levels "P01","P02","P03",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ fca8      : Factor w/ 51 levels "117/133","119/133",..: NA NA 35 26 26 35 31 35 41 31 ...
 $ fca23     : Factor w/ 39 levels "128/130","128/136",..: 23 39 23 26 31 23 23 23 23 12 ...
 $ fca43     : Factor w/ 28 levels "133/133","133/135",..: 15 17 19 16 21 27 17 9 15 21 ...
 $ fca45     : Factor w/ 32 levels "116/116","116/118",..: 3 13 1 4 22 13 4 13 4 13 ...
 $ fca77     : Factor w/ 48 levels "132/154","142/142",..: 44 44 38 30 36 33 36 43 35 33 ...
 $ fca78     : Factor w/ 25 levels "138/138","140/140",..: 11 11 8 11 11 22 11 11 8 22 ...
 $ fca90     : Factor w/ 39 levels "181/185","181/199",..: 37 9 35 37 30 28 37 29 35 35 ...
 $ fca96     : Factor w/ 40 levels "091/091","091/101",..: 34 34 34 4 34 8 20 1 20 4 ...
 $ fca37     : Factor w/ 39 levels "182/182","182/192",..: 25 25 34 25 25 25 25 27 27 25 ...
 - attr(*, "locicol")= int  2 3 4 5 6 7 8 9 10
```

هر یک از اجزای شیء loci با استفاده از علامت $ قابل استخراج است. مثلاً برای مشاهده اطلاعات جمعیت شیء l.cats، می‌توان دستور زیر را اجرا کرد:

```
> l.cats$population
  [1] P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P02 P02 P02 P02 P02 P02 P02 P02
 [19] P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P03 P03 P03 P03
……..
[217] P16 P12 P12 P12 P12 P12 P12 P12 P17 P17 P17 P17 P17 P17 P17 P17 P17 P17
[235] P17 P17 P17
17 Levels: P01 P02 P03 P04 P05 P06 P07 P08 P09 P10 P11 P12 P13 P14 P15 ... P17
```

همچنین به عنوان مثال، اطلاعات لوکوس fca8 با دستور زیر قابل مشاهده است:

```
> l.cats$fca8
  [1] <NA>    <NA>    135/143 133/135 133/135 135/143 135/135 135/143 137/143
 [10] 135/135 137/141 129/133 129/133 133/133 131/135 129/135 129/133 129/135
…….
```

[226] 133/135 133/141 133/141 123/133 123/133 133/141 133/141 133/141 133/143

[235] 135/141 137/143 135/141

51 Levels: 117/133 119/133 121/135 121/137 121/145 123/123 123/133 ... 149/149

سطر آخر خروجی دستور فوق (51 Levels) نشان می‌دهد که در لوکوس fca8، ۵۱ ژنوتیپ متفاوت قابل شناسایی می‌باشد. با استفاده از تابع getGenotypes می‌توان انواع ژنوتیپ‌ها را برای تمام لوکوس‌ها به تفکیک، مشاهده نمود:

> getGenotypes(l.cats)

$fca8

[1] "117/133" "119/133" "121/135" "121/137" "121/145" "123/123" "123/133"

[8] "123/135" "123/137" "123/139" "123/141" "123/143" "127/133" "129/129"

……..

[50] "145/145" "149/149"

$fca23

[1] "128/130" "128/136" "128/146" "128/150" "130/130" "130/136" "130/140"

……..

[29] "140/140" "140/144" "140/146" "140/148" "140/150" "142/142" "142/144"

[36] "142/146" "144/144" "144/146" "146/146"

……..

$fca37

[1] "182/182" "182/192" "182/194" "182/202" "182/204" "182/206" "182/208"

…….

[36] "210/214" "212/212" "214/214" "214/218"

ژنوتیپ‌های یک لوکوس خاص با ترکیب دستور فوق و علامت $ قابل استخراج است مثلاً برای لوکوس fca90 خواهیم داشت:

```
> getGenotypes(l.cats)$fca90
 [1] "181/185" "181/199" "185/185" "185/189" "185/191" "185/193" "185/195"
……
[29] "193/197" "193/199" "193/201" "195/195" "195/197" "195/199" "197/197"
[36] "197/199" "199/199" "199/203" "199/205"
```

اسامی آلل‌ها به تفکیک لوکوس‌های مختلف را می‌توان با تابع getAlleles بدست آورد:

```
> getAlleles(l.cats)
$fca8
 [1] "117" "133" "119" "121" "135" "137" "145" "123" "139" "141" "143" "127"
[13] "129" "131" "147" "149"

$fca23
 [1] "128" "130" "136" "146" "150" "140" "142" "144" "132" "138" "148"
……..

$fca37
 [1] "182" "192" "194" "202" "204" "206" "208" "216" "224" "184" "186" "200"
[13] "214" "220" "226" "210" "212" "218"
```

در اینجالت نیز انواع آلل‌های یک لوکوس خاص با ترکیب دستور فوق و علامت $ قابل استخراج است. مثلاً برای لوکوس fca90 خواهیم داشت:

```
> getAlleles(l.cats)$fca90
 [1] "181" "185" "199" "189" "191" "193" "195" "197" "201" "203" "187" "205"
```

مقادیر فراوانی‌های ژنوتیپی و آللی به ازای لوکوس‌های مختلف با استفاده از تابع summary در پکیج pegas قابل مشاهده می‌باشد:

```
> s.l.cats <-summary(l.cats)
> s.l.cats
Locus fca8 :
```

-- Genotype frequencies:

117/133 119/133 121/135 121/137 121/145 123/123 123/133 123/135 123/137 123/139

1 1 2 3 1 1 2 6 10 1

123/141 123/143 127/133 129/129 129/131 129/133 129/135 129/137 129/141 129/145

6 2 1 2 4 4 3 2 2 1

……..

-- Allele frequencies:

117	119	121	123	127	129	131	133	135	137	139	141	143	145	147	149
1	1	6	29	1	20	22	33	105	83	27	41	44	11	3	7

……..

خروجی تابع summary از نوع لیست است (لیستی از لوکوس‌ها به همراه اسامی ژنوتیپ و آلل‌ها و مقادیر فراوانی آن‌ها) و اجزاء این لیست با علامت $ قابل استخراج می‌باشند:

```
> str(s.l.cats)
```

List of 9

$ fca8 :List of 2

..$ genotype: Named int [1:51] 1 1 2 3 1 1 2 6 10 1 ...

.. ..- attr(*, "names")= chr [1:51] "117/133" "119/133" "121/135" "121/137" ...

..$ allele : Named num [1:16] 1 1 6 29 1 20 22 33 105 83 ...

.. ..- attr(*, "names")= chr [1:16] "117" "119" "121" "123" ...

$ fca23:List of 2

..$ genotype: Named int [1:39] 1 3 1 1 8 16 3 1 1 8 ...

.. ..- attr(*, "names")= chr [1:39] "128/130" "128/136" "128/146" "128/150" ...

..$ allele : Named num [1:11] 6 51 24 172 24 47 20 16 101 4 ...

.. ..- attr(*, "names")= chr [1:11] "128" "130" "132" "136" ...

……

```
> names(s.l.cats)
```

[1] "fca8" "fca23" "fca43" "fca45" "fca77" "fca78" "fca90" "fca96" "fca37"

به عنوان مثال برای مشاهده فراوانی‌های ژنوتیپی و آللی لوکوس fca8، می‌توان دستورات زیر را اجرا نمود:

```
> s.l.cats$fca8
```

$genotype

117/133 119/133 121/135 121/137 121/145 123/123 123/133 123/135 123/137 123/139

```
      1         1         2         3         1         1         2         6        10         1
……..

 137/143   137/147   139/139   139/143   141/141   141/145   143/143   143/145   143/147   145/145
       8         1         6         3         7         2         5         2         1         1
 149/149
       3
          $allele

  17  119  121  123  127  129  131  133  135  137  139  141  143  145  147  149
   1    1    6   29    1   20   22   33  105   83   27   41   44   11    3    7
```

هریک از فراوانی‌های ژنوتیپی و آللی برای لوکوس fca8، با دستورات جداگانه نیز قابل استخراج است. بدین منظور دستورات زیر را اجرا و نتایج را مشاهده نمایید:

```
> s.l.cats$fca8$genotype
```

```
> s.l.cats$fca8$allele
```

محاسبه فراوانی آللی نسبی

با استفاده از تابع apply به شرح زیر می‌توان فراوانی نسبی آلل‌ها در لوکوس‌های مختلف را محاسبه نمود. در این دستور آرگومان na.rm=TRUE سبب می‌شود که مقادیر گمشده قبل از محاسبه میانگین، حذف شوند و عدد ۲ سبب اعمال شدن میانگین بر روی ستون‌ها (فراوانی آلل‌ها) می‌شود:

apply(tab(x, freq=TRUE), 2, mean, na.rm=TRUE)

به عنوان مثال فراوانی نسبی آلل‌ها در لوکوس‌های مختلف از مجموعه داده nancycats بصورت زیر قابل محاسبه می‌باشد:

```
> library(adegenet)
```

```
> data(nancycats)
```

```
> x <- nancycats
```

```
> apply(tab(x,  freq=TRUE), 2, mean,  na.rm=TRUE)
```

fca8.117	**fca8.119**	**fca8.121**	**fca8.123**	**fca8.127**	**fca8.129**
0.002304	0.002304	0.013825	0.06682	0.002304	0.046083
fca8.131	**fca8.133**	**fca8.135**	**fca8.137**	**fca8.139**	**fca8.141**
0.050691	0.076037	0.241935	0.191244	0.062212	0.09447
fca8.143	**fca8.145**	**fca8.147**	**fca8.149**	**fca23.128**	**fca23.130**
0.101382	0.025346	0.006912	0.016129	0.012658	0.107595

……

fca37.202	fca37.204	fca37.206	fca37.208	fca37.210	fca37.212
0.008439	0.012658	0.084388	0.607595	0.023207	0.010549
fca37.214	**fca37.216**	**fca37.218**	**fca37.220**	**fca37.224**	**fca37.226**
0.037975	0.014768	0.004219	0.010549	0.004219	0.004219

محاسبه فراوانی آللی بر اساس افراد و جمعیت‌ها

تابع makefreq(x) از پکیج adegenet برای محاسبه فراوانی آللی نسبی در سطح افراد و جمیعت‌ها قابل استفاده می‌باشد. در این تابع x، می‌تواند یک شیء genind یا genpop باشد. در حالت genind، داده‌ها در سطح فرد حفظ می‌شوند، اما استاندارد شده و حاصل جمع آنها برابر با یک می‌باشد. به عنوان مثال با دستورات زیر می‌توان فراوانی آللی نسبی را در افراد مجموعه داده microbov بدست آورد:

```
> library(adegenet)
> data(microbov)
> obj1 <- microbov
> xfreq1 <- makefreq(obj1)
> tab(microbov)[1:20,1:5]
```

	INRA63.167	INRA63.171	INRA63.173	INRA63.175	INRA63.177
AFBIBOR9503	0	0	0	0	0
AFBIBOR9504	0	0	0	0	0
AFBIBOR9505	0	0	0	0	1
AFBIBOR9506	0	0	0	0	0
AFBIBOR9507	0	0	0	0	1
AFBIBOR9508	0	0	0	0	1
AFBIBOR9509	0	0	0	0	1
AFBIBOR9510	0	0	0	0	0
AFBIBOR9511	0	0	0	0	1
AFBIBOR9512	0	0	0	0	0
AFBIBOR9513	0	0	0	0	2
AFBIBOR9514	0	0	0	0	0
AFBIBOR9515	0	0	0	0	0
AFBIBOR9516	0	0	0	0	1
AFBIBOR9517	0	0	0	0	1
AFBIBOR9518	0	0	0	1	0
AFBIBOR9519	0	0	0	0	1
AFBIBOR9520	0	0	0	0	0
AFBIBOR9521	0	0	0	0	0
AFBIBOR9522	0	0	0	0	2

```
> xfreq1[1:20,1:5]
```

	INRA63.167	INRA63.171	INRA63.173	INRA63.175	INRA63.177
AFBIBOR9503	0	0	0	0	0
AFBIBOR9504	0	0	0	0	0
AFBIBOR9505	0	0	0	0	0.5
AFBIBOR9506	0	0	0	0	0
AFBIBOR9507	0	0	0	0	0.5
AFBIBOR9508	0	0	0	0	0.5
AFBIBOR9509	0	0	0	0	0.5
AFBIBOR9510	0	0	0	0	0
AFBIBOR9511	0	0	0	0	0.5
AFBIBOR9512	0	0	0	0	0
AFBIBOR9513	0	0	0	0	1
AFBIBOR9514	0	0	0	0	0
AFBIBOR9515	0	0	0	0	0
AFBIBOR9516	0	0	0	0	0.5
AFBIBOR9517	0	0	0	0	0.5
AFBIBOR9518	0	0	0	0.5	0
AFBIBOR9519	0	0	0	0	0.5
AFBIBOR9520	0	0	0	0	0
AFBIBOR9521	0	0	0	0	0
AFBIBOR9522	0	0	0	0	1

با مقایسه نتایج دو دستور tab(microbov)[1:20,1:5] (مربوط به فراوانی مطلق یا همان عدد خام حاصل از شمارش تعداد آلل‌ها) و xfreq1[1:20,1:5] (مربوط به فراوانی نسبی آلل‌ها) در فوق، می‌توانید اثر تابع makefreq را بر روی شیء genind (در مجموعه داده microbov) بررسی نمایید.

اثر تابع makefreq را بر روی شیء genpop (جمعیت‌ها بجای افراد) در مثال زیر قابل مشاهده و بررسی می‌باشد:

```
> obj2 <- genind2genpop(microbov)
> xfreq2 <- makefreq(obj2)
> tab(obj2)[,1:5]
```

	INRA63.167	INRA63.171	INRA63.173	INRA63.175	INRA63.177
Borgou	0	0	0	4	27
Zebu	0	1	0	7	16
Lagunaire	1	0	0	16	44
NDama	0	0	0	2	39

Somba	0	0	0	12	42
Aubrac	0	0	0	80	0
Bazadais	0	0	0	54	28
BlondeAquitaine	0	0	0	54	52
BretPieNoire	0	0	1	39	18
Charolais	0	0	5	46	37
Gascon	0	0	0	77	1
Limousin	0	0	1	45	52
MaineAnjou	0	1	0	46	48
Montbeliard	0	0	0	25	25
Salers	0	0	0	70	0

> xfreq2[,1:5]

	INRA63.167	INRA63.171	INRA63.173	INRA63.175	INRA63.177
Borgou	0.0000000	0.0000000	0.0000000	0.0400000	0.2700000
Zebu	0.0000000	0.0100000	0.0000000	0.0700000	0.1600000
Lagunaire	0.0098039	0.0000000	0.0000000	0.1568628	0.4313725
NDama	0.0000000	0.0000000	0.0000000	0.0333333	0.6500000
Somba	0.0000000	0.0000000	0.0000000	0.1200000	0.4200000
Aubrac	0.0000000	0.0000000	0.0000000	0.8000000	0.0000000
Bazadais	0.0000000	0.0000000	0.0000000	0.5869565	0.3043478
BlondeAquitaine	0.0000000	0.0000000	0.0000000	0.4500000	0.4333333
BretPieNoire	0.0000000	0.0000000	0.0161290	0.6290323	0.2903226
Charolais	0.0000000	0.0000000	0.0480769	0.4423077	0.3557692
Gascon	0.0000000	0.0000000	0.0000000	0.7700000	0.0100000
Limousin	0.0000000	0.0000000	0.0100000	0.4500000	0.5200000
MaineAnjou	0.0000000	0.0104167	0.0000000	0.4791667	0.5000000
Montbeliard	0.0000000	0.0000000	0.0000000	0.4166667	0.4166667
Salers	0.0000000	0.0000000	0.0000000	0.7000000	0.0000000

در دستورات فوق tab(obj2)[,1:5]، فراوانی مطلق آلل‌ها و xfreq2[,1:5] فراوانی نسبی آلل‌ها را در جمعیت‌هایِ مجموعه داده microbov نشان می‌دهد.

محاسبه فراوانی آلل فرعی (MAF)

منظور از فراوانی آلل فرعی[۱۳] (MAF)، فراوانی مربوط به دومین آللِ معمول در جمعیت است. این فراوانی بطور گسترده‌ای در مطالعات ژنتیک جمعیت استفاده می‌شود زیرا با استفاده از آن می‌توان واریانت‌های

[13] Minor Allele Frequency

معمول و نادر در جمعیت را تشخصیص داد. با استفاده از تابع minorAllele در پکیج adegenet می‌توان فراوانی آلل فرعی (minor) را، در هر لوکوس بدست آورد که در آن x، یک شیء genind است. به عنوان مثال فراوانی آلل فرعی به تفکیک لوکوس‌ها در مجموعه داده nancycats به شرح زیر قابل مشاهده می‌باشد:

```
> library(adegenet)
> data(nancycats)
> x <- nancycats
> minorAllele(x)
     fca8      fca23      fca43      fca45      fca77      fca78      fca90      fca96
0.1912442 0.2130802 0.2046414 0.1944444 0.1751055 0.1645570 0.2278481 0.3114035
     fca37
0.1139241
```

رسم نمودار ستونی فراوانی آلل‌ها

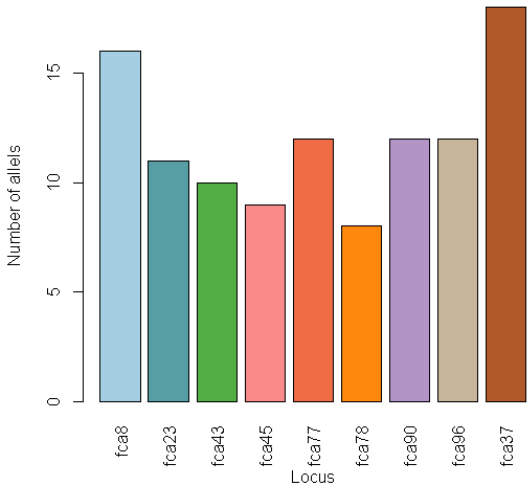
همانطور که قبلاً اشاره شد با استفاده از @all.names برای یک شیء genind، می‌توان تعداد آلل‌ها را به تفکیک لوکوس‌ها بدست آورد. با استفاده از این اطلاعات به عنوان ورودی تابع barplot، می‌توان نمودار ستونی فراوانی‌های آللی در لوکوس‌های متفاوت را ترسیم نمود. به عنوان مثال برای مجموعه داده nancycats نمودار ستونی فراوانی‌های آللی به ازای لوکوس‌های مختلف بصورت زیر قابل ترسیم می‌باشد (شکل ۴-۸):

```
> library(adegenet)
> data(nancycats)
> barplot(nancycats@loc.n.all, col=funky(9), las=3, xlab="Locus", ylab="Number of allels")
```

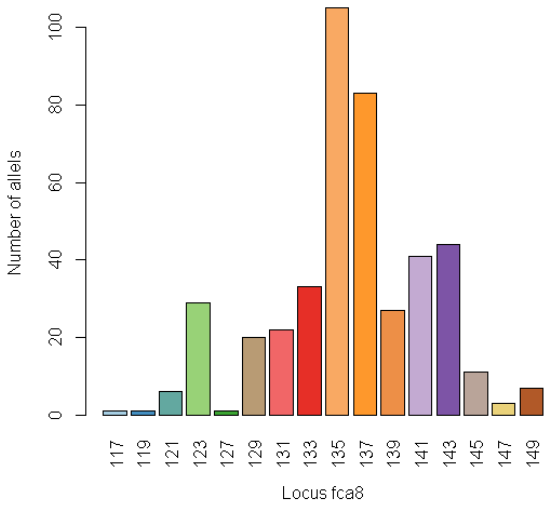
همچنین می‌توان فراوانی آللی در هر لوکوس را با نمودار ستونی نمایش داد. به عنوان مثال برای نمایش فراوانی آللی در لوکوس fca8 با استفاده از نمودار ستونی، می‌توان دستور زیر را اجرا نمود (شکل ۴-۹):

```
> library(pegas)
> barplot(summary(genind2loci(nancycats))$fca8$ allele, col=funky(16), las=3, xlab="Locus fca8", ylab="Number of allels")
```

جهت توضیح پیرامون دستور summary(genind2loci(nancycats)) که به عنوان آرگومان در تابع barplot استفاده شده است، به بخش محاسبه فراوانی‌های ژنوتیپی و آللی مراجعه نمایید.



شکل ۴-۸- نمودار ستونی فراوانی آلل‌ها به تفکیک لوکوس‌ها در مجموعه داده nancycats



شکل ۴-۹- نمودار ستونی فراوانی آلل‌ها برای لوکوس fca8 در مجموعه داده nancycats

محاسبه تعداد انواع ژنوتیپ‌های مورد انتظار

با استفاده از تابع expand.genotype در پکیج pegas، می‌توان تعداد ژنوتیپ‌های مورد انتظار را برحسب تعداد آلل‌های مشاهده شده در هر لوکوس، به دست آورد:

```
> library(pegas)
> expand.genotype(2)
     [1] "1/1" "1/2" "2/2"
```

در دستور فوق، عدد داخل پرانتز تعداد آلل‌ها را مشخص می‌سازد و با ذکر عدد ۲، دو آللِ 1 و 2 برای موجود دیپلوئید (پیش‌فرض) در نظر گرفته می‌شود. به همین ترتیب برای سه یا چهار آلل خواهیم داشت:

```
> expand.genotype(3)
     [1] "1/1" "1/2" "1/3" "2/2" "2/3" "3/3"
> expand.genotype(4)
     [1] "1/1" "1/2" "1/3" "1/4" "2/2" "2/3" "2/4" "3/3" "3/4" "4/4"
```

می‌توان برای آلل‌ها بجای اعداد، کاراکترهای حرفی در نظر گرفت:

```
> expand.genotype(3, c("A","B","C"))
     [1] "A/A" "A/B" "A/C" "B/B" "B/C" "C/C"
```

برای سهولت انجام کار، می‌توان از وکتورهای از پیش‌تعریف شده‌ی letters یا LETTERS برای درج کاراکترهای حرفی استفاد کرد. در این وکتورها، حروف براساس ترتیب الفبا ذخیره شده‌اند، به مثال‌های زیر توجه نمایید:

```
> letters[1:5]
     [1] "a" "b" "c" "d" "e"
> LETTERS[1:5]
     [1] "A" "B" "C" "D" "E"
> length(LETTERS)
     [1] 26
```

```
> LETTERS[13:21]
 [1] "M" "N" "O" "P" "Q" "R" "S" "T" "U"
```

به عنوان مثال می‌توان وکتور LETTERS را برای نشان دادن آلل‌ها با حروف A، B و C بصورت زیر در تابع expand.genotype به کار برد:

```
> expand.genotype(3, LETTERS[1:3])
 [1] "A/A" "A/B" "A/C" "B/B" "B/C" "C/C"
```

با استفاده از تابع expand.genotype همچنین می‌توان انواع و تعداد ژنوتیپ‌ها را در سطوح پلوئیدی متفاوت بدست آورد:

```
> expand.genotype(3, ploidy = 4)
 [1] "1/1/1/1" "1/1/1/2" "1/1/1/3" "1/1/2/2" "1/1/2/3" "1/1/3/3" "1/2/2/2"
 [8] "1/2/2/3" "1/2/3/3" "1/3/3/3" "2/2/2/2" "2/2/2/3" "2/2/3/3" "2/3/3/3"
[15] "3/3/3/3"
> length(expand.genotype(4, ploidy = 3))
 [1] 20
```

تابع expand.genotype برای کنترل درجه آزادی آزمون χ^2 مربوط به برقراری تعادل هاردی واینبرگ نیز مفید است که در بخش مربوطه توضیح داده خواهد شد. در اینجا برای تمرین ابتدا تعداد انواع آلل‌ها و ژنوتیپ‌های مشاهده شده را برای لوکوس fca90 از مجموعه داده nancycats بدست می‌آوریم، سپس براساس تعداد آلل‌ها در این لوکوس، تعداد ژنوتیپ‌های مورد انتظار را محاسبه می‌کنیم:

```
> library(adegenet)
> library(pegas)
> data(nancycats)
> l.cats<-genind2loci(nancycats)
> n.all.fca90<-length(getAlleles(l.cats)$fca90)
> n.all.fca90
 [1] 12
> n.gen.fca90<-length(getGenotypes(l.cats)$fca90)
> n.gen.fca90
```

```
[1] 39
> length(expand.genotype(n.all.fca90, ploidy = 2))
[1] 78
```

نتایج فوق نشان می‌دهد که براساس تعداد انواع آلل‌ها (۱۲ آلل) در لوکوس fca90 ، تعداد ۷۸ نوع ژنوتیپ مورد انتظار است، اما فقط ۳۹ نوع ژنوتیپ در این مجموعه داده، مشاهده شده است.

محاسبه فراوانی‌های ژنوتیپی مورد انتظار

تابع pgen از پکیج poppr، فراوانی ژنوتیپی مورد انتظار را براساس حاصلضرب فراوانی آللی در هر لوکوس و برای هر فرد، با فرض برقرار بودن تعادل هاردی واینبرگ محاسبه می‌نماید. به عنوان مثال فراوانی‌های ژنوتیپی مورد انتظار در مجموعه داده nancycats به شرح زیر قابل محاسبه می‌باشد:

```
> library(adegenet)
> library(poppr)
> data(nancycats)
> head(pgen(nancycats, log = FALSE))
```

	fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
N215	NA	0.27	0.1225	0.15	0.1225	0.5	0.16	0.3	0.64
N216	NA	0.2025	0.175	0.225	0.1225	0.5	0.04	0.3	0.64
N217	0.28125	0.27	0.0625	0.09	0.175	0.25	0.1225	0.3	0.01
N218	0.14063	0.01	0.175	0.27	0.0625	0.5	0.16	0.1	0.64
N219	0.14063	0.045	0.125	0.2025	0.0625	0.5	0.12	0.3	0.64
N220	0.28125	0.27	0.05	0.225	0.175	0.25	0.015	0.28	0.64

محاسبه میزان اشتراک آللی

تابع propShared از پکیج adgenet، نسبت آلل‌های مشترک در مجموعه‌ای از ژنوتیپ‌ها که در قالب شیء genind ذخیره شده باشند را محاسبه می‌کند.

برای مثال لوکوس‌های اول و دوم از مجموعه داده microbov را برای پنج فرد اول جدا می‌کنیم و نسبت آلل‌های مشترک در آنها را به دست می‌آوریم:

```
> library(adegenet)
```

```
> data(microbov)
> obj <- microbov[1:5, loc = locNames(microbov)[1:2]]
> propShared(obj)
```

	AFBIBOR9503	AFBIBOR9504	AFBIBOR9505	AFBIBOR9506	AFBIBOR9507
AFBIBOR9503	1	0.5	0.5	0.75	0.5
AFBIBOR9504	0.5	1	0.75	0.75	0.75
AFBIBOR9505	0.5	0.75	1	0.75	1
AFBIBOR9506	0.75	0.75	0.75	1	0.75
AFBIBOR9507	0.5	0.75	1	0.75	1

برای درک بهتر نتایج، می‌توان با استفاده از دستور زیر، آلل‌ها را در لوکوس‌های مورد نظر، مشاهده و با نتایج فوق تطبیق داد:

```
> genind2df(obj,sep="/")
```

	pop	INRA63	INRA5
AFBIBOR9503	Borgou	183/183	137/141
AFBIBOR9504	Borgou	181/183	141/141
AFBIBOR9505	Borgou	177/183	141/141
AFBIBOR9506	Borgou	183/183	141/141
AFBIBOR9507	Borgou	177/183	141/141

همانطور که در جدول فوق مشخص است، در دو فرد AFBIBOR9503 و AFBIBOR9504، مجموعاً چهار نوع آلل (137، 141، 181 و 183) برای دو لوکوس INRA63 و INRA5 مشاهده می‌شود که از لحاظ دو آلل (141 و 183) مشترک می‌باشند، لذا میزان اشتراک آللی ۵۰ درصد (یا ۰/۵) می‌باشد. توجه داشته باشید که میزان اشتراک آللی می‌تواند به عنوان معیاری از شباهت ژنتیکی افراد (و عکس آن به عنوان کمیّتی از فاصله بین آنها) درنظر گرفته شود.

شناسایی آلل‌های اختصاصی

تابع private_alleles از پکیج poppr، آلل‌هایی که فقط در یک جمعیت ظاهر شده‌اند را شناسایی می‌نماید. به عنوان مثال آلل‌های اختصاصی برای جمعیت‌های مجموعه داده nancycats را می‌توان با دستور زیر شناسایی نمود:

```
> library(adegenet)
```

```
> library(poppr)
> data(nancycats)
> private_alleles(nancycats)[,1:6]
```

	fca8.117	fca8.119	fca8.127	fca43.157	fca77.132	fca78.138
P01	0	0	0	0	0	0
P02	0	0	0	0	0	0
P03	0	0	0	0	0	0
……						
P14	1	1	1	0	0	0
P15	0	0	0	0	0	0
P16	0	0	0	0	0	0
P17	0	0	0	0	0	0

نتایج فوق نشان می‌دهد که به عنوان مثال در جمعیت شماره ۱۴ در لوکوس (fca8) سه نوع آلل (fca8.117، fca8.119 و fca8.127) وجود دارد که در سایر جمعیت‌ها مشاهده نشده است. به همین ترتیب می‌توان آلل‌های اختصاصی برای هر نوع دسته بندی در داده‌ها را شناسایی نمود. به عنوان مثال آلل‌های اختصاصی برای مجموعه داده microbov، پس از انتساب دسته‌بندی (با استفاده از تابع strata که قبلاً شرح داده شد)، برحسب کشور، نژاد و گونه به شرح زیر قابل شناسایی می‌باشند:

```
> data(microbov)
> strata(microbov) <- data.frame(other(microbov))
> private_alleles(microbov, alleles ~ coun)[,1:7]
```

	INRA63.167	INRA63.173	INRA63.181	INRA5.137	INRA5.149	ETH225.153	ETH225.155
AF	1	0	47	29	1	3	4
FR	0	7	0	0	0	0	0

```
> private_alleles(microbov, alleles ~ breed)[,1:6]
```

	INRA63.167	INRA5.149	HEL5.169	ETH152.181	ETH152.187	ETH152.205
Borgou	0	0	0	0	0	0
Zebu	0	1	0	0	0	0
Lagunaire	1	0	0	0	0	0
Limousin	0	0	0	0	1	0
……..						
Montbeliard	0	0	0	0	0	0
Salers	0	0	0	1	0	0

```
> private_alleles(microbov, alleles ~ spe)[,1:7]
```

	INRA63.167	INRA63.173	INRA5.147	INRA5.149	ETH225.137	ETH225.151	ETH225.153
BI	0	0	0	1	0	0	3
BT	1	7	3	0	4	13	0

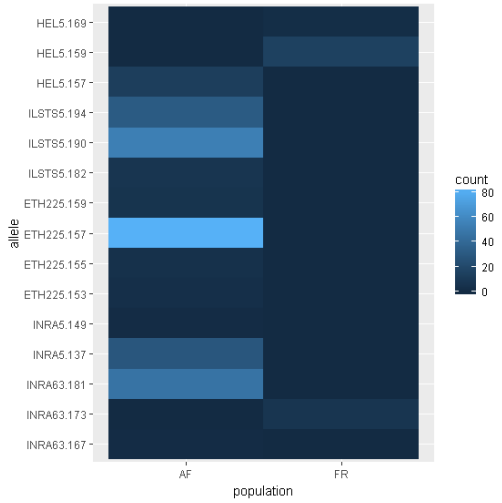
نمایش تعداد آلل‌های اختصاصی به صورت نمودار، امکان مقایسه چشمی دستجات و وضعیت کلی آلل‌های متمایز کننده‌ی گروه‌ها را فراهم می‌کند. به عنوان مثال نمودار آلل‌های اختصاصی برای پنج لوکوس اول در مجموعه داده microbov به ترتیب برای کشور و گونه با استفاده از دستورات زیر قابل ترسیم می‌باشد (شکل‌های ۴-۱۰ و ۴-۱۱):

توجه: پکیج ggplot2 را باید از قبل نصب نموده باشید.

```
> library(adegenet)
> library(poppr)
> library(ggplot2)
> data(microbov)
> strata(microbov) <- data.frame(other(microbov))
> loc.mic<- microbov[loc= locNames(microbov)[1:5]]
> mic.pri.count <- private_alleles(loc.mic, alleles ~ coun, report = "data.frame")
> head(mic.pri.count)
```

	population	allele	count
1	AF	INRA63.167	1
2	FR	INRA63.167	0
3	AF	INRA63.173	0
4	FR	INRA63.173	7
5	AF	INRA63.181	47
6	FR	INRA63.181	0

```
> ggplot(mic.pri.count) + geom_tile(aes(x = population, y = allele, fill = count))
```

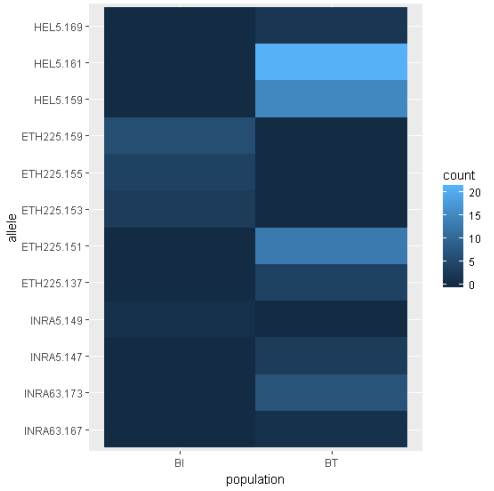


شکل ۴-۱۰–مقایسه فراوانی آلل‌های اختصاصی در دسته‌بندی جمعیت‌ها براساس کشور از مجموعه داده microbov

```
> mic.pri.spe <- private_alleles(loc.mic, alleles ~ spe, report = "data.frame")
> head(mic.pri.spe)
```

	population	allele	count
1	BI	INRA63.167	0
2	BT	INRA63.167	1
3	BI	INRA63.173	0
4	BT	INRA63.173	7
5	BI	INRA5.147	0
6	BT	INRA5.147	3

```
> ggplot(mic.pri.spe) + geom_tile(aes(x = population, y = allele, fill = count))
```



شکل ۴-۱۱–مقایسه فراوانی آلل‌های اختصاصی در دسته‌بندی جمعیت‌ها براساس گونه از مجموعه داده microbov

هتروزیگوسیتی مورد انتظار و آزمون تفاوت

محاسبه هتروزیگوسیتی مورد انتظار یکی از رایج‌ترین روش‌های برآورد تنوع درون جمعیتی است (Nybom, 2004). تابع Hs در پکیج adgenet، هتروزیگوسیتی مورد انتظار را برای جمعیت‌های موجود در اشیاء genind یا genpop، محاسبه می‌کند. این تابع فقط برای نشانگرهای همبارز (@type="codom") قابل کاربرد می‌باشد.

پارامتر Hs معمولاً به عنوان معیار اندازه‌گیری تنوع ژنتیکی درون جمعیت، مورد استفاده قرار می‌گیرد. چنانچه $m(k)$ تعداد آلل‌های لوکوس K، و f_i فراوانی آلل iام در جمعیت مورد نظر باشد، هتروزیگوسیتی مورد انتظار (Hs) برابر خواهد بود با:

$$\frac{1}{K}\sum_{k=1}^{K}(1-\sum_{i=1}^{m(k)}f_i^2)$$

به عنوان مثال هتروزیگوسیتی مورد انتظار برای جمعیت‌های مجموعه داده nancycats به شرح زیر قابل مشاهده می‌باشد:

```
> library(adegenet)
> data(nancycats)
> Hs(nancycats)
```

P01	P02	P03	P04	P05	P06	P07	P08
0.615729	0.685262	0.689043	0.733459	0.619012	0.711203	0.641006	0.714444
P09	P10	P11	P12	P13	P14	P15	P16
0.655693	0.666667	0.761478	0.642592	0.661407	0.763826	0.690083	0.671296
P17							
0.625628							

برای آزمون تفاوت معنی‌دار بین هتروزیگوسیتی مورد انتظار دو جمعیت، می‌توان از تابع Hs.test (از پکیج adegenet) استفاده کرد. این آزمون به روش مونت کارلو و از طریق انتساب تصادفی افراد به جمعیت‌های مورد آزمون و بدست آوردن توزیع مرجع برای آماره آزمون، انجام می‌شود:

Hs.test(x, y, n.sim = 999, alter = c("two-sided", "greater", "less"))

x و y، جمعیت‌های شیء genind هستند که تفاوت هتروزیگوسیتی مورد انتظار در آنها، مورد آزمون می‌باشد. n.sim، تعداد جایگشت‌ها[۱۴] برای ایجاد توزیع مرجع است. با استفاده از آرگومان alter می‌توان مشخص کرد که آزمون به‌صورت دوطرفه (two-sided) یا به‌صورت یکطرفه (greater برای بزرگتر و less برای کوچکتر) انجام شود.

به عنوان مثال، برای بررسی تفاوت معنی‌دار هتروزیگوسیتی مورد انتظار، بین جمعیت‌های Borgou و Lagunaire، از مجموعه داده microbov، به شرح زیر عمل می‌شود:

```
> data(microbov)
> Hs(microbov)
```

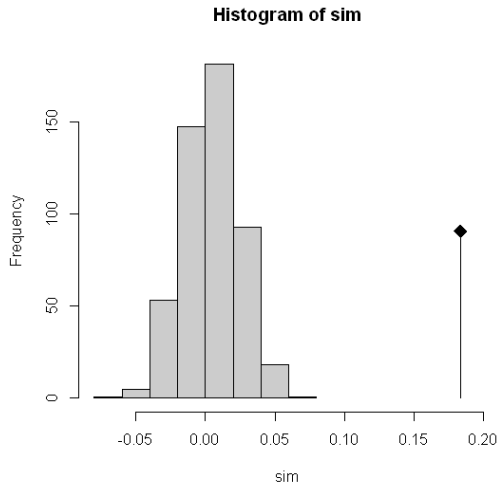
Borgou	**Zebu**	**Lagunaire**	**NDama**	**Somba**
0.722688	0.7058975	0.5389179	0.6043297	0.6123879
Aubrac	**Bazadais**	**BlondeAquitaine**	**BretPieNoire**	**Charolais**
0.6551573	0.5860082	0.6782948	0.6823884	0.6796183
Gascon	**Limousin**	**MaineAnjou**	**Montbeliard**	**Salers**
0.6897316	0.6677523	0.6096112	0.6663166	0.6054594

```
> test <- Hs.test(microbov[pop="Borgou"], microbov[pop="Lagunaire"], n.sim=499)
> test

        Monte-Carlo test

        Call: Hs.test(x = microbov[pop = "Borgou"], y = microbov[pop = "Lagunaire"],     n.sim = 499)

        Observation: 0.1837701

        Based on 499 replicates

        Simulated p-value: 0.002

        Alternative hypothesis: two-sided

          Std.Obs.y       Expectation        Variance
        8.8274431051  0.0045957559  0.0004119855
```

همانطور که مشاهده می‌شود p-value از مقدار کوچکی ($P<0.01$) برخوردار است که حاکی از معنی‌دار بودن تفاوت هتروزیگوسیتی مورد انتظار، بین جمعیت‌های Borgou و Lagunaire می‌باشد. این تفاوت معنی‌دار را می‌توان با ترسیم نمودار توزیع فراوانی، نیز نشان داد (شکل ۴-۱۲):

[14] Permutations

> plot(test)



شکل ۴-۱۲– نمودار توزیع مرجع برای آزمون تفاوت هتروزیگوسیتی مورد انتظار، بین جمعیت‌های Borgou و Lagunaire در مجموعه داده microbov به روش مونت کارلو

مقایسه هتروزیگوتی مشاهده شده و مورد انتظار

مقایسه مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار اطلاعات مفیدی را حاصل نماید از جمله اینکه می‌تواند نشانه‌ای از وجود درون‌زادآوری[۱۵] باشد. برای به دست آوردن مقادیر هتروزیگوتی مشاهده شده و مورد انتظار به تفکیک لوکوس‌ها، می‌توان از تابع summary در پکیج adgenet استفاده نمود. از آنجا که پکیج pegas نیز دارای تابعی با نام مشابه (summary) می‌باشد، به منظور اینکه برای R مشخص نماییم که تابع summary از پکیج adgenet را اجرا نماید، از علامت :: استفاده می‌نماییم. به عنوان مثال دو سطر آخر خروجی تابع summary برای مجموعه داده nancycats، به ترتیب هتروزیگوسیتی مشاهده شده و مورد انتظار را به تفکیک لوکوس‌ها نشان می‌دهد:

```
> library(adegenet)
>  data(nancycats)
> adegenet::summary(nancycats)
    // Number of individuals: 237
    // Group sizes: 10 22 12 23 15 11 14 10 9 11 20 14 13 17 11 12 13
    // Number of alleles per locus: 16 11 10 9 12 8 12 12 18
```

[15] Inbreeding

// Number of alleles per group: 36 53 50 67 48 56 42 54 43 46 70 52 44 61 42 40 35

// Percentage of missing data: 2.34 %

// Observed heterozygosity: 0.67 0.67 0.68 0.71 0.63 0.57 0.65 0.62 0.45

// Expected heterozygosity: 0.87 0.79 0.8 0.76 0.87 0.69 0.82 0.76 0.61

خروجی دستور فوق از نوع لیست است و مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار به ازای لوکوس‌های مختلف با استفاده از علامت $ قابل استخراج می‌باشد.

مقادیر هتروزیگوسیتی مشاهده شده در لوکوس‌های مختلف:

```
> obs.cats<-adegenet::summary(nancycats)$Hobs
> obs.cats
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
0.668	0.667	0.679	0.708	0.633	0.565	0.65	0.618	0.451

مقادیر هتروزیگوسیتی مورد انتظار در لوکوس‌های مختلف:

```
> exp.cats<-adegenet::summary(nancycats)$Hexp
> exp.cats
```

fca8	fca23	fca43	fca45	fca77	fca78	fca90	fca96	fca37
0.866	0.793	0.795	0.76	0.87	0.688	0.816	0.76	0.606

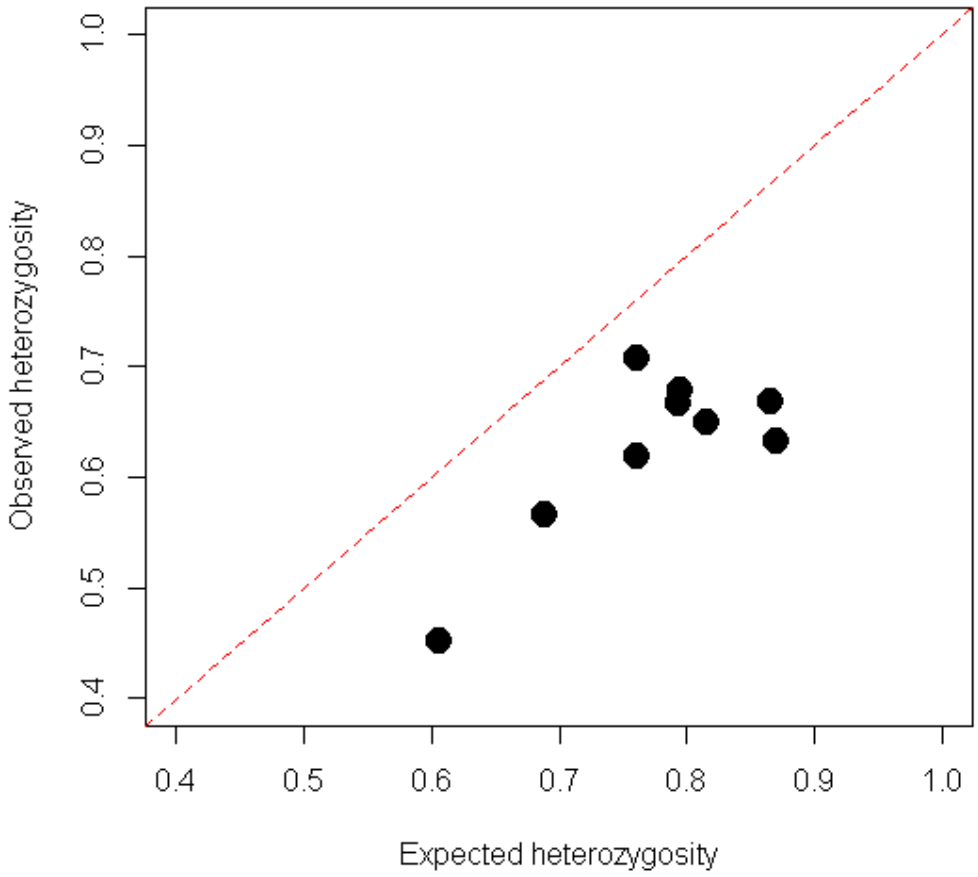
با مقایسه مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار بصورت چشمی، مشاهده می‌شود که در تمام لوکوس‌ها، مقادیر مشاهده شده کوچکتر از مقادیر مورد انتظار می‌باشند. این نتایج می‌توان نشانه‌ای از رخداد پدیده درون‌زادآوری باشد که در نتیجه آن، جمعیت از حالت آمیزش تصادفی منحرف می‌شود. برای آنکه تصویر واضح‌تری از این مقایسه بدست آید، می‌توان این مقادیر را در برابر یکدیگر، بصورت نمودار نمایش داد (شکل ۴-۱۳):

```
> plot(exp.cats, obs.cats, pch=20, cex=3, xlim=c(.4,1), ylim=c(.4,1), xlab="Expected heterozygosity", ylab="Observed heterozygosity")
> abline(0,1,lty=2, col="red")
```

در دستورات فوق آرگومان pch، به کاراکتر درج شده در نمودار اشاره دارد که مقدار ۲۰ نشان دهنده دایره توپُر می‌باشد. می‌توان دستور را با مقادیر ۱۷، ۱۸، ۲۱ و ... برای آرگومان مذکور، مجدداً اجرا و تفاوت در

نتایج را مشاهده نمود. آرگومان lty نیز به نوع خط اشاره دارد. برای توضیح بیشتر در مورد اعداد اختصاصی مربوط به خطوط و علایم به بخش رسم نمودار از فصل دوم این کتاب (آشنایی با محیط و انواع متغیرها در R) مراجعه نمایید. آرگومان cex به اندازه کاراکترها اشاره دارد و عدد ۳، نشان‌دهنده‌ی اندازه سه مرتبه بزرگتر از مقدار پیش‌فرض است. آرگومان‌های xlim و ylim به محدوده نمایش داده شده در محور مختصات افقی و عمودی اشاره دارد، این آرگومان‌ها برای محدود کردن فضای نمودار به حوالی مختصات نقاط پراکنش و اجتناب از فضای خالی بیهوده در نمودار، مفید می‌باشند.

تابع abline برای درج خط مرجع در نمودار مورد استفاده قرار می‌گیرد. اعداد صفر و یک در این تابع به ترتیب به عبور خط از مبدا مختصات و شیب آن اشاره دارد.



شکل ۴-۱۳– نمودار پراکنش لوکوس‌ها برحسب مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار درمجموعه داده nancycats

اگرچه مقایسه مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار به صورت چشمی، کوچکتر بودن مقادیر مشاهده شده از مقادیر مورد انتظار را نشان می‌دهد، اما این مشاهده ظاهری، برای نتیجه‌گیری کافی نیست و تفاوت مذکور باید مورد آزمون آماری قرار گیرد. بنابراین دو گروه از اعداد داریم که می‌توانیم تفاوت آماری آنها را با آزمون t بررسی نماییم. اما برای انجام این آزمون، نخست باید از یکنواختی واریانس دو گروه مطمئن شویم. بدین منظور آزمون بارتلت را انجام می‌دهیم:

```
> bartlett.test(list(exp.cats, obs.cats))

        Bartlett test of homogeneity of variances

data:  list(exp.cats, obs.cats)
Bartlett's K-squared = 0.046962, df = 1, p-value = 0.8284
```

مقدار p-value آزمون بسیار بزرگ است که حاکی از یکنواختی واریانس دو گروه است. اکنون می‌توان آزمون t را انجام داد:

```
> t.test(exp.cats, obs.cats, pair=T,var.equal=TRUE,alter="greater")

        Paired t-test

data:  exp.cats and obs.cats
t = 8.3294, df = 8, p-value = 1.631e-05
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 0.1134779       Inf
sample estimates:
mean of the differences
             0.1460936
```

در تابع فوق آرگومان pair=T، به مقایسات جفتی اشاره دارد. توجه داشته باشید از آنجا که مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار مربوط به مجموعه لوکوس‌های مشترکی می‌باشند (یعنی برای هر لوکوس، مقدار هتروزیگوسیتی مشاهده شده و هم مقدار هتروزیگوسیتی مورد انتظار محاسبه می‌شود) بنابراین آزمون از نوع جفتی می‌باشد. آرگومان var.equal=TRUE به یکنواختی واریانس دو گروه اشاره دارد

که در مرحله قبل، آزمون شد و وجود یکنواختی تأیید گردید. آرگومان "alter="greater یکطرفه بودن آزمون، براساس بزرگتر بودن مقادیر هتروزیگوسیتی مورد انتظار از مشاهده شده را تعیین می‌نماید.

مقدار کوچک p-value نشان می‌دهد که بین مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار، تفاوت معنی‌داری وجود دارد. توجه داشته باشید که درجه آزادی df = 8، مربوط به مقایسه ۹ لوکوس می‌باشد. برای احراز اطمینان از این تعداد، می‌توان دستور زیر را وارد اجرا نمود:

```
> nLoc(nancycats)
    [1] 9
```

به عنوان تمرین، تفاوت مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار را در بزرگترین جمعیت از مجموعه داده nancycats بررسی نمایید. بدین منظور ابتدا لازم است اندازه جمیعت‌های تشکیل دهنده این مجموعه داده را بررسی نماییم:

```
> table(pop(nancycats))
```

P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	P16	P17
10	22	12	23	15	11	14	10	9	11	20	14	13	17	11	12	13

مشاهده می‌شود که جمعیت چهارم از بیشترین افراد (۲۳ فرد) تشکیل شده است. اکنون می‌توان با استفاده از تابع seppop، جمعیت‌ها را در قالب لیست، از یکدیگر جدا کرد و سپس با استفاده از علامت $، جمعیت چهارم را فراخواند:

```
> p.cats<-seppop(nancycats)
> class(p.cats)
[1] "list"
> names(p.cats)
    [1] "P01" "P02" "P03" "P04" "P05" "P06" "P07" "P08" "P09" "P10" "P11" "P12"
    [13] "P13" "P14" "P15" "P16" "P17"
> p4.cats<-p.cats$P04
> p4.cats
    /// GENIND OBJECT /////////
     // 23 individuals; 9 loci; 108 alleles; size: 24.8 Kb
     // Basic content
```

@tab: 23 x 108 matrix of allele counts

@loc.n.all: number of alleles per locus (range: 8-18)

@loc.fac: locus factor for the 108 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: .local(x = x, i = i, j = j, treatOther = ..1, quiet = ..2, drop = drop)

// Optional content

@pop: population of each individual (group size range: 23-23)

@other: a list containing: xy

```
> nInd(p4.cats)
```

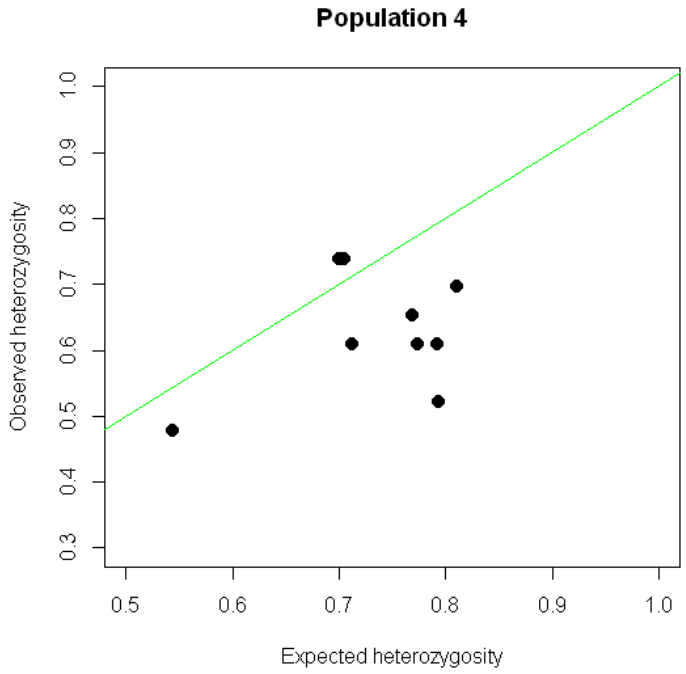
[1] 23

توجه داشته باشید عمل تمایزِ جمعیتِ چهارم، با دستور nancycats[pop=4] که قبلاً شرح داده شد نیز قابل انجام است، ولی در این تمرین به منظور آشنایی بیشتر از تابع seppop استفاده شد.

```
> s.p4.cats<-summary(p4.cats)
```

اکنون نمودار مقادیر مشاهده شده در برابر مقادیر مورد انتظار هتروزیگوسیتی به تفکیک لوکوس‌ها در جمعیت چهارم از مجموعه داده nancycats با دستورات زیر قابل ترسیم خواهد بود (شکل ۴-۱۴):

```
> plot(s.p4.cats$Hexp,s.p4.cats$Hobs, xlim=c(0.5,1), ylim=c(0.3,1), pch=20, cex=2, xlab="Expected heterozygosity", ylab="Observed heterozygosity", main="Population 4")

> abline(0,1,col="green")
```



شکل ۴-۱۴– نمودار پراکنش لوکوس‌ها برحسب مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار در جمعیت چهارم از مجموعه داده nancycats

همچنین آزمون تفاوت معنی‌دار بین مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار برای لوکوس‌های جمعیت چهارم از مجموعه داده nancycats به شرح زیر قابل انجام است:

```
> bartlett.test(list(s.p4.cats$Hexp,s.p4.cats$Hobs))

        Bartlett test of homogeneity of variances

data:  list(s.p4.cats$Hexp, s.p4.cats$Hobs)
Bartlett's K-squared = 0.056569, df = 1, p-value = 0.812


> t.test(s.p4.cats$Hexp,s.p4.cats$Hobs,pair=T,var.equal=TRUE,alter="greater")

        Paired t-test

data:  s.p4.cats$Hexp and s.p4.cats$Hobs
t = 3.1738, df = 8, p-value = 0.00656
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 0.04366203        Inf
```

sample estimates:

mean of the differences

0.10544

مقدار کوچک p-value نشان می‌دهد که در جمعیت چهارم بین مقادیر هتروزیگوسیتی مشاهده شده و مورد انتظار، تفاوت بسیار معنی‌داری وجود دارد ($P<0.01$)، و به معنی کوچکتر بودن مقادیر هتروزیگوسیتی مشاهده شده از مورد انتظار و شاهدی از رخداد درون‌زادآوری در این جمعیت می‌باشد.

محتوای اطلاعات پلی‌مورفیسم (PIC)

بطور کلی میزان پلی‌مورفیسم توسط دو معیار کمّیِ هتروزیگوسیتی و محتوای اطلاعات پلی‌مورفیسم (PIC) قابل اندازه‌گیری می‌باشد. PIC توسط بوتستین (Botstein *et al.*, 1980) برای نشانگرهای همبارز معرفی شد و اما بعدها برای نشانگرهای غالب نیز مورد استفاده قرار گرفت (Shete *et al.*, 2000). فرمول محاسباتی PIC مشابه با هتروزیگوسیتی مورد انتظار به ازای هر لوکوس است. درنشانگرهایی مانند میکروستلایت‌ها که نسبت یک به یک بین نشانگر و لوکوس واقع است و به عبارت دیگر هر نشانگر (عمدتاً) یک لوکوس را هدف قرار می‌دهد، می‌توان همان برآورد هتروزیگوسیتی مورد انتظار بدست آمده از تابع summary در پکیج adgenet را به عنوان PIC، مد نظر قرار دارد. با اینحال محاسبه PIC به راحتی امکان‌پذیر است. در اینجا دو روش فوق را برای مجموعه داده nancycats مقایسه می‌نماییم:

```
> library(adegenet)
>  data(nancycats)
> locNames(nancycats)
[1] "fca8"  "fca23" "fca43" "fca45" "fca77" "fca78" "fca90" "fca96" "fca37"
> summary(nancycats)$Hexp
     fca8      fca23      fca43      fca45      fca77      fca78      fca90      fca96      fca37
0.865722   0.792875   0.795332   0.760310   0.870258   0.688467   0.815788   0.760349   0.606269
```

نتایج فوق، هتروزیگوسیتی مورد انتظار برای ۹ لوکوس متناظر با ۹ نشانگر میکروستلایت را نشان می‌دهد. اکنون می‌توان PIC را به عنوان مثال برای fca8 و fca37 محاسبه نمود:

```
> l.cats<-genind2loci(nancycats)

> fca8.no.alle<-summary(l.cats)$fca8$allele

> fca8.no.alle

 117  119  121  123  127  129  131  133  135  137  139  141  143  145  147  149
   1    1    6   29    1   20   22   33  105   83   27   41   44   11    3    7

> fca8.freq.alle<-fca8.no.alle/sum(fca8.no.alle)

> PIC.fca8<-1-sum(fca8.freq.alle^2)

> PIC.fca8

[1] 0.8657224

> fca37.no.alle<-summary(l.cats)$fca37$allele

> fca37.freq.alle<-fca37.no.alle/sum(fca37.no.alle)

> PIC.fca37<-1-sum(fca37.freq.alle^2)

> PIC.fca37

[1] 0.6062686
```

مشاهده می‌شود که مقادیر بدست آمده با مقادیر هتروزیگوسیتی مورد انتظار برای لوکوس‌های مربوطه، در تطابق می‌باشد.

آزمون برقراری تعادل هاردی واینبرگ

آزمون انحراف از نسبت‌های هاردی واینبرگ به منظور بررسی آمیزش تصادفی در جمعیت‌ها انجام می‌شود و انحراف از نسبت‌های مورد انتظار برای برآورد ضریب درون‌زادآوری مورد استفاده قرار می‌گیرد (Robertson and Hill, 1984). این آزمون با استفاده از تابع hw.test در پکیج pegas و به دو صورت قابل انجام می‌باشد. شیوه اول، همان آزمون χ^2 مرسوم است که مبتنی بر فراوانی‌های ژنوتیپی مورد انتظارِ محاسبه شده از فراوانی‌های آللی می‌باشد. روش دوم آزمونی دقیق، براساس جایگشت‌های رویه مونت کارلو برای آلل‌ها است. آرگومان B در تابع hw.test تعیین می‌کند که کدام روش اجرا شود. آرگومان B، عبارت از تعداد تکرارها در رویه مونت کارلو است. اگر B، برابر صفر قرار داده شود (B=0)، آزمون کلاسیک χ^2 انجام می‌شود. توجه داشته باشید که روش مونت کارلو با استفاده از تابع hw.test، فقط برای دیپلوئیدها قابل انجام است.

```
> library(adegenet)
> library(pegas)
> data(nancycats)
> cats.hwt<- hw.test(nancycats, B=0)
> cats.hwt
```

	chi^2	df	Pr(chi^2>)
fca8	395.8001	120	0.00 e+00
fca23	239.3422	55	0.00 e+00
fca43	434.334	45	0.00 e+00
fca45	66.11849	36	1.62 e-03
fca77	270.5207	66	0.00 e+00
fca78	402.8	28	0.00 e+00
fca90	217.1984	66	0.00 e+00
fca96	193.3676	66	1.97e-14
fca37	291.0073	153	1.21e-10

نتایج فوق نشان می‌دهد که در تمام لوکوس‌ها انحراف نسبت‌های ژنوتیپی مشاهده شده از مورد انتظار، معنی‌دار می‌باشد و به عبارت دیگر تعادل هاردی واینبرگ برقرار نیست. این آزمون را برای هر یک از جمعیت‌های مجموعه داده nancycats نیز می‌توان انجام داد که لازمه‌ی آن، ابتدا جدا کردن جمعیت مورد نظر در یک شیء genind مجزا و سپس انجام آزمون است که نحوه انجام این کار قبلاً در بخشِ مقایسه هتروزیگوتی مشاهده شده و مورد انتظار، شرح داده شد و لذا بدلیل پرهیز از از اطاله کلام، به توضیح آن نمی‌پردازیم.

توجه: همانطور که مشاهده می‌شود در آزمون فوق، درجه آزادی برای لوکوس‌های گوناگون، متفاوت است و دلیل آن هم، وجود تعداد آلل‌های متفاوت به ازای هر لوکوس می‌باشد، که در نتیجه متناظر با آن، تعداد ژنوتیپ‌ها برای هر لوکوس، متفاوت خواهد بود. برای کنترل کردن درجه آزادی نیاز به محاسبه تعداد ژنوتیپ‌هایِ ممکن براساس تعداد آلل‌های در هر لوکوس موجود می‌باشد. این کار با استفاده از تابع expand.genotype در پکیج pegas قابل انجام است که نحوه انجام آن در بخش بعدی توضیح داده می‌شود.

برای ذخیره خروجی دستورات می‌توان از تابع write.table استفاده نمود. برای اینکار ابتدا بهتر است ابتدا اعداد را (مثلاً تا سه رقم بعد از اعشار) گرد کنیم.

```
> r.cats<-round(cats.hwt,3)
```

```
> r.cats
```

	chi^2	df	Pr(chi^2>)
fca8	395.800	120	0.000
fca23	239.342	55	0.000
fca43	434.334	45	0.000
fca45	66.118	36	0.002
fca77	270.521	66	0.000
fca78	402.800	28	0.000
fca90	217.198	66	0.000
fca96	193.368	66	0.000
fca37	291.007	153	0.000

```
> write.table(r.cats, file = "D:/HW.txt", sep = "\t")
```

با اجرای دستور فوق، فایل HW با فرمت متنی (.txt) در درایو :D ذخیره می‌شود. آرگومان "\t" = sep تعیین می‌کند که در فایل خروجی، ستون‌ها با tab از یکدیگر جدا شوند.

اکنون می‌توانیم با استفاده از تابع expand.genotype (که در بخش محاسبه تعداد انواع ژنوتیپ‌های مورد انتظار در فصل چهارم شرح داده شد) درجه آزادی آزمون χ^2 در بخش قبل را برای لوکوس‌های مختلف کنترل نماییم. به عنوان مثال تعداد انواع آلل‌های مشاهده شده برای لوکوس fca8 عبارتست از:

```
> n.all.fca8<-length(getAlleles(as.loci(nancycats))$fca8)
> n.all.fca8
[1] 16
```

انواع ممکن ژنوتیپ‌ها (مورد انتظار) را با داشتن ۱۶ آلل، با استفاده از تابع expand.genotype بدست می‌آوریم:

```
> n.gen.fca8<-length(expand.genotype(n.all.fca8))
> n.gen.fca8
[1] 136
```

درجه آزادی آزمون χ^2 عبارت از k-n می‌باشد که در آن k، تعداد کل انواع ژنوتیپ‌های مورد انتظار، و n، تعداد انواع آلل‌های مشاهده شده است:

```
> df.fca8<-n.gen.fca8-n.all.fca8
> df.fca8
[1] 120
```

در اینجا به عنوان تمرین درجه آزادی سه لوکوس دیگر را کنترل می‌نماییم. درجه آزادی آزمون χ^2 برای لوکوس fca23:

```
> n.all.fca23<-length(getAlleles(as.loci(nancycats))$fca23)
> n.all.fca23
[1] 11
> n.gen.fca23<-length(expand.genotype(n.all.fca23))
> n.gen.fca23
[1] 66
> df.fca23<-n.gen.fca23-n.all.fca23
> df.fca23
[1] 55
```

درجه آزادی آزمون χ^2 برای لوکوس fca43:

```
> n.all.fca43<-length(getAlleles(as.loci(nancycats))$fca43)
> n.all.fca43
[1] 10
> n.gen.fca43<-length(expand.genotype(n.all.fca43))
> n.gen.fca43
[1] 55
> df.fca43<-n.gen.fca43-n.all.fca43
> df.fca43
[1] 45
```

درجه آزادی آزمون χ^2 برای لوکوس fca45:

```
>  n.all.fca45<-length(getAlleles(as.loci(nancycats))$fca45)
> n.all.fca45
[1] 9
> n.gen.fca45<-length(expand.genotype(n.all.fca45))
```

```
> n.gen.fca45
        [1] 45
> df.fca45<-n.gen.fca45-n.all.fca45
> df.fca45
        [1] 36
```

فصل پنجم- فواصل ژنتیکی

محاسبه فواصل ژنتیکی بین جمعیت‌ها

توابع فواصل ژنتیکی در پکیج adegenet

تابع dist.genpop در پکیج adegenet، فاصله ژنتیکی بین جمعیت‌ها را محاسبه می‌کند:

dist.genpop(x, method = 1)

که در آن، شیئی x از کلاس genpop است و method، روش محاسبه فواصل می‌باشد که چنانچه ذکر نشود روش اول به عنوان پیش‌فرض، مدنظر قرار می‌گیرد. درحال حاضر محاسبه پنج معیار فاصله به شرح ذیل توسط این تابع امکان‌پذیر است که برخی از آنها از نوع اقلیدسی می‌باشند.

۱) فاصله Nei (غیراقلیدسی) (Nei, 1972؛ Nei, 1978؛ Avise, 1994).

۲) فاصله زاویه‌ای یا فاصله Edwards (اقلیدسی) (Edwards, 1971؛ Cavalli-Sforza and Edwards, 1967؛ Hartl and Clark, 1997).

۳) ضریب جد مشترک یا فاصله Reynolds (اقلیدسی) (Reynolds *et al.*,1983).

۴) فاصله کلاسیک اقلیدسی یا فاصله Rogers (اقلیدسی) (Rogers, 1972؛ Avise, 1994).

۵) فاصله ژنتیکی مطلق یا فاصله Provesti (غیراقلیدسی) (Prevosti ,1974؛ Prevosti *et al.*, 1975).

مثال: می‌خواهیم در مجموعه داده nancycats، فواصل بین جمعیت‌ها را محاسبه کنیم. ابتدا با استفاده از تابع genind2genpop مجموعه داده nancycats که از نوع genind است را به شیء genpop تبدیل می‌کنیم:

```
> library(adegenet)
> data(nancycats)
> p.cats<-genind2genpop(nancycats)
```

سپس با استفاده از تابع dist.genpop به روش شماره یک (فاصله Nei)، فواصل بین جمعیت‌ها را به دست می‌آوریم:

```
> d1.p.cats<-dist.genpop(p.cats, met=1)
> class(d1.p.cats)
        [1] "dist"
```

همانطور که مشاهده می‌شود کلاسِ متغیر حاصل شده از نوع فاصله است. برای مشاهده فواصل، می‌توان شیء حاصله را به ماتریس تبدیل کرد:

```
> as.matrix(d1.p.cats)[1:5,1:5]
```

	P01	P02	P03	P04	P05
P01	0.000000	0.477272	0.337322	0.292186	0.232072
P02	0.477272	0.000000	0.560793	0.453920	0.434151
P03	0.337322	0.560793	0.000000	0.154504	0.234592
P04	0.292186	0.453920	0.154504	0.000000	0.179356
P05	0.232072	0.434151	0.234592	0.179356	0.000000

همانطور که در فوق اشاره شد، فاصله Nei از نوع غیراقلیدسی است. این موضوع را می‌توان با تابع is.euclid امتحان کرد:

```
> is.euclid(d1.p.cats)
        [1] FALSE
```

اکنون با استفاده از تابع dist.genpop به روش شماره ۲ (فاصله Edwards)، فواصل ژنتیکی را به دست می‌آوریم:

```
> d2.p.cats<-dist.genpop(p.cats, met=2)
```

همانطور که در فوق اشاره شد، فاصله Edwards از نوع اقلیدسی است:

```
> is.euclid(d2.p.cats)
        [1] TRUE
```

توجه داشته باشید که با استفاده از تابع cailliez می‌توان فواصل غیراقلیدسی را به فواصل اقلیدسی تبدیل کرد. اثر این تابع را بر روی فواصل بدست آمده از روش اول (که غیراقلیدسی بودند) را می‌توان امتحان نمود:

```
> d1c.p.cats<-cailliez(d1.p.cats)
> is.euclid(d1c.p.cats)
      [1] TRUE
```

توابع فواصل ژنتیکی در پکیج hierfstat

با استفاده از تابع genet.dist در پکیج hierfstat می‌توان انواع فاصله ژنتیکی بین جمعیت‌ها را محاسبه نمود:

genet.dist(dat ,diploid=TRUE, method="Dch")

dat: یک قالب داده است که در آن، اسامی جمعیت‌ها در ستون اول و لوکوس‌ها در ستون‌های بعدی آمده است.

diploid: این آرگومان یک گزاره منطقی است که مشخص می‌کند آیا موجود دیپلوئید است یا خیر. این گزینه بطور پیش‌فرض، دیپلوئید تعیین شده است.

method: این آرگومان روش محاسبه فاصله ژنتیکی را تعیین می‌کند. یکی از گزینه‌های Dch، Da، Ds، Fst، Dm، Dr، Cp و X2 را می‌توان برای این آرگومان انتخاب کرد. این روش‌ها توسط تاکزاکی و نی (Takezaki and Nei, 1996) شرح داده شده‌اند.

روش Dch، مربوط به فاصله Cavalli-Sforza و Edwards Chord می‌باشد که در معادله شماره ۶ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996) ذکر شده است. تاکزاکی و نی (Takezaki and Nei, 1996) این فاصله را بهترین روش تعیین رابطه بین نمونه‌ها دانستند، لذا روش Dch، به عنوان گزینه پیش‌فرض برای آرگومان method در نظر گرفته شده است.

روش Da، مربوط به فاصله ژنتیکی Nei و همکاران است می‌باشد که در معادله شماره ۷ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996) ذکر شده است. این روش نیز کارایی مشابه با Dch را دارد.

گزینه Ds، به فاصله ژنتیکی استاندارد نی در معادله شماره ۱ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996) اشاره دارد که بصورت خطی افزایش می‌یابد، ولی واریانس بزرگتری دارد.

گزینه Fst، فاصله ژنتیکی Reynolds و همکاران مندرج در معادله شماره ۳ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996) را محاسبه می‌کند.

گزینه Dm: حداقل فاصله Nei، مندرج در معادله شماره ۲ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996).

گزینه Dr: فاصله Rogers، مندرج در معادله شماره ۴ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996).

گزینه Cp: فاصله Prevosti، مندرج در معادله شماره ۵ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996).

گزینه X2: فاصله Sanghvi، مندرج در معادله شماره ۸ از مرجع تاکزاکی و نی (Takezaki and Nei, 1996).

بعلاوه، گزینه‌های Nei87 و WC84 را نیز می‌توان مورد استفاده قرار داد که برآورد Fstهای جفتی بین جمعیت‌ها را به ترتیب براساس روش نی (Nei, 1987) و ویر و کوکرام (Weir and Cockerham, 1984) محاسبه می‌کند.

به عنوان مثال فواصل ژنتیکی را در مجموعه داده gtrunchier به دست می‌آوریم:

```
> library(adegenet)
> library(hierfstat)
> data(gtrunchier)
> head(gtrunchier,10)
```

	Locality	Patch	L21.V	L37.J	L20.B	L29.V	L36.B	L16.J
1	1	1	22	11	11	11	55	55
2	1	1	77	11	55	11	22	55
3	1	1	22	11	55	22	55	55
4	1	1	77	11	55	11	22	55
5	1	1	27	23	55	22	22	57
6	1	1	22	11	55	22	55	55
7	1	1	22	11	11	11	55	55
8	1	1	77	22	55	11	22	77
9	1	1	22	11	55	22	22	55
10	1	1	77	22	55	11	22	77

```
> as.matrix(genet.dist(gtrunchier[,-1]))[1:10,1:7]
```

	1	2	3	4	5	6	7
1	0.000000	0.048559	0.495445	0.558306	0.408464	0.745101	0.751720
2	0.048559	0.000000	0.495297	0.569944	0.295573	0.732990	0.739559
3	0.495445	0.495297	0.000000	0.139560	0.569353	0.568650	0.564912
4	0.558306	0.569944	0.139560	0.000000	0.587370	0.499630	0.492508
5	0.408464	0.295573	0.569353	0.587370	0.000000	0.548792	0.522512
6	0.745101	0.732990	0.568650	0.499630	0.548792	0.000000	0.020377
7	0.751720	0.739559	0.564912	0.492508	0.522512	0.020377	0.000000
8	0.745101	0.732990	0.556336	0.482215	0.509852	0.038941	0.014391
9	0.745101	0.732990	0.559459	0.486632	0.519727	0.010064	0.007412

10	0.751720	0.739559	0.562636	0.489290	0.515317	0.046136	0.007196

```
> as.matrix(genet.dist(gtrunchier[,-1] ,method="Dr"))[1:10,1:7]
```

	1	2	3	4	5	6	7
1	0.000000	0.164619	0.386597	0.421353	0.441869	0.564497	0.577557
2	0.164619	0.000000	0.424547	0.463895	0.297713	0.576583	0.589944
3	0.386597	0.424547	0.000000	0.233695	0.567549	0.504640	0.514719
4	0.421353	0.463895	0.233695	0.000000	0.599956	0.434041	0.452096
5	0.441869	0.297713	0.567549	0.599956	0.000000	0.510684	0.461393
6	0.564497	0.576583	0.504640	0.434041	0.510684	0.000000	0.055556
7	0.577557	0.589944	0.514719	0.452096	0.461393	0.055556	0.000000
8	0.586828	0.599557	0.524439	0.453891	0.455128	0.055556	0.022222
9	0.578936	0.591461	0.517075	0.446097	0.470280	0.040404	0.015152
10	0.583459	0.595995	0.520245	0.457958	0.450282	0.066667	0.011111

محاسبه فواصل ژنتیکی بین افراد

با استفاده از تابع diss.dist در پکیج poppr می‌توان فاصله ژنتیکی بین افراد را محاسبه نمود. خروجی این تابع بطور ساده تفاوت آللی بین افراد می‌باشد:

diss.dist(x, percent = FALSE, mat = FALSE)

که در آن x ، شیئی از کلاس genind است. آرگومان منطقی percent تعیین می‌کند که آیا فواصل به صورت درصد ارائه شوند یا خیر. برای محاسبه فواصل برحسب درصد، مقادیر تفاوت آللی بر حاصلضرب سطح پلوئیدی در تعداد لوکوس، تقسیم می‌شود. همچنین آرگومان منطقی mat تعیین می‌کند که آیا خروجی تابع از نوع ماتریس باشد یا خیر.

به عنوان مثال فاصله ژنتیکی بین افراد در جمعیت هشتم از مجموعه داده nancycats با استفاده از تابع diss.dist بصورت زیر قابل محاسبه است:

```
> library(adegenet)
>  data(nancycats)
> diss.dist(popsub(nancycats, 8))
```

	N43	N92	N94	N95	N96	N98	N99	N100	N93
N92	14								
N94	15	15							
N95	11	15	13						
N96	14	12	12	12					
N98	12	13	11	11	14				

N99	14	13	11	17	14	10			
N100	13	16	12	14	15	9	10		
N93	10	15	10	12	12	11	11	13	
N97	10	13	14	11	15	5	13	12	10

باید توجه داشت در صورت انتخاب گزینه TRUE برای آرگومان percent، فواصل محاسبه شده، مشابه با فواصل بدست آمده از تابع provesti.dist خواهد بود:

```
> round(diss.dist(popsub(nancycats, 8), percent =TRUE),4)
```

	N43	N92	N94	N95	N96	N98	N99	N100	N93
N92	0.7778								
N94	0.8333	0.8333							
N95	0.6111	0.8333	0.7222						
N96	0.7778	0.6667	0.6667	0.6667					
N98	0.6667	0.7222	0.6111	0.6111	0.7778				
N99	0.7778	0.7222	0.6111	0.9444	0.7778	0.5556			
N100	0.7222	0.8889	0.6667	0.7778	0.8333	0.5	0.5556		
N93	0.5556	0.8333	0.5556	0.6667	0.6667	0.6111	0.6111	0.7222	
N97	0.5556	0.7222	0.7778	0.6111	0.8333	0.2778	0.7222	0.6667	0.5556

همچنین در پکیج poppr مجموعه‌ای از توابع به شرح زیر برای محاسبه فواصل ژنتیکی بین افراد براساس روش‌های نی، ادوادز، راجرز، رینولدز و پروستی (Nei, 1972، Nei, 1978، Avise, 1994، Edwards, 1971، Rogers, 1972، Reynolds *et al.*, 1983، Hartl and Clark, 1989، Cavalli-Sforza and Edwards, 1967، Prevosti, 1974، Prevosti and Alonso, 1975، Gower and Legendre, 1986 و Legendre and Legendre, 1998) موجود می‌باشد:

nei.dist(x)

edwards.dist(x)

rogers.dist(x)

reynolds.dist(x)

prevosti.dist(x)

به عنوان مثال با استفاده از توابع فوق، فواصل ژنتیکی بین افراد را در جمعیت پنجم از مجموعه داده nancycats محاسبه می‌نماییم.

```
> library(adegenet)
> library(poppr)
```

```
> data(nancycats)

> p5.cats <- popsub(nancycats, 5)

> nei.cats <- nei.dist(p5.cats)

> class(nei.cats)

[1] "dist"

> as.matrix(nei.cats)[1:5,1:5]
```

	N55	N56	N57	N58	N59
N55	0.000000	0.646674	1.251628	1.132182	0.505801
N56	0.646674	0.000000	0.861383	0.441833	0.626382
N57	1.251628	0.861383	0.000000	0.966744	0.928149
N58	1.132182	0.441833	0.966744	0.000000	0.834021
N59	0.505801	0.626382	0.928149	0.834021	0.000000

```
> edw.cats <- edwards.dist(p5.cats)

> as.matrix(edw.cats)[1:5,1:5]
```

	N55	N56	N57	N58	N59
N55	0.000000	0.697510	0.836188	0.787809	0.646130
N56	0.697510	0.000000	0.736258	0.593630	0.745356
N57	0.836188	0.736258	0.000000	0.751723	0.758036
N58	0.787809	0.593630	0.751723	0.000000	0.751723
N59	0.646130	0.745356	0.758036	0.751723	0.000000

```
> rog.cats <- rogers.dist(p5.cats)

> as.matrix(rog.cats)[1:5,1:5]
```

	N55	N56	N57	N58	N59
N55	0.000000	0.523012	0.627363	0.622009	0.508126
N56	0.523012	0.000000	0.563681	0.444444	0.463468
N57	0.627363	0.563681	0.000000	0.581339	0.636895
N58	0.622009	0.444444	0.581339	0.000000	0.578567
N59	0.508126	0.463468	0.636895	0.578567	0.000000

```
> rey.cats <- reynolds.dist(p5.cats)

> as.matrix(rey.cats)[1:5,1:5]
```

	N55	N56	N57	N58	N59
N55	0.000000	0.722315	0.707107	0.779194	0.741620
N56	0.722315	0.000000	0.733799	0.745356	0.836660
N57	0.707107	0.733799	0.000000	0.745356	0.784465
N58	0.779194	0.745356	0.745356	0.000000	0.859727
N59	0.741620	0.836660	0.784465	0.859727	0.000000

```
> pre.cats <- prevosti.dist(p5.cats)
> as.matrix(pre.cats)[1:5,1:5]
```

	N55	N56	N57	N58	N59
N55	0.000000	0.555556	0.722222	0.666667	0.555556
N56	0.555556	0.000000	0.611111	0.444444	0.555556
N57	0.722222	0.611111	0.000000	0.611111	0.666667
N58	0.666667	0.444444	0.611111	0.000000	0.611111
N59	0.555556	0.555556	0.666667	0.611111	0.000000

ترسیم فواصل ژنتیکی

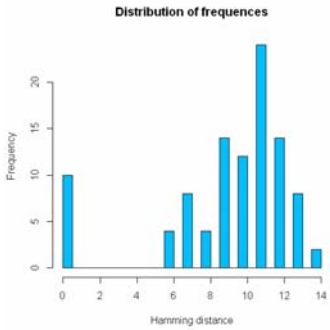
تابع gengraph از پکیج adegenet نموداری براساس فواصل ژنتیکی ایجاد می‌کند و درآن زوج موجودیت‌ها (افراد یا جمعیت‌ها) اگر و فقط اگر دارای فاصله‌ای کمتر از یک آستانه تعیین شده باشند، به یکدیگر متصل می‌شوند. برای اینکار از الگوریتم‌های نمودار و کلاس‌های موجود در پکیج igraph استفاده می‌شود:

gengraph(x, cutoff=NULL, col.pal=funky)

این تابع، ماتریس و اشیاء dist، genind، genpop و DNAbin را به عنوان ورودی (x) می‌پذیرد. به عنوان مثال در اولین جمعیت از مجموعه داده nancycats، فواصل ژنتیکی را رسم می‌کنیم:

```
> library(adegenet)
> library(poppr)
>  data(nancycats)
> g <- gengraph(popsub(nancycats, 1))
```

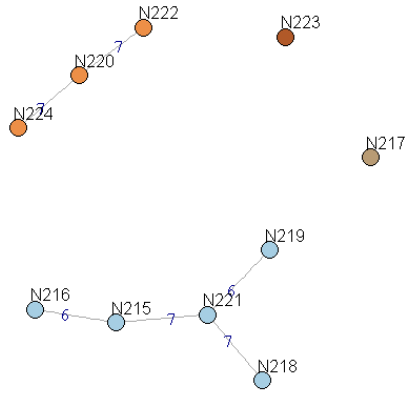
از آنجا که در تابع فوق آستانه فاصله ژنتیکی برای رسم نمودار توسط آرگومان cutoff، تعیین نشده است، نموداری از تفاوت‌های آللی بین افراد نمایش داده شده (شکل ۵-۱) و از کاربر خواسته می‌شود حد آستانه را تعیین کند:



شکل ۵-۱– توزیع فراوانی فواصل ژنتیکی بین افراد در اولین جمعیت از مجموعه داده nancycats

به عنوان مثال با تعیین فاصله ژنتیکی با مقدار (آستانه) ۸، نمودار شکل ۵-۲ ترسیم می‌شود:

```
Please choose a cutoff distance:  8

Number of clusters found:  4
Are you satisfied with this solution? (yes:y / no:n): y
```



شکل ۵-۲– نمایش گرافیکی فواصل ژنتیکی بین افراد در اولین جمعیت از مجموعه داده nancycats با لحاظ کردن مقدار ۸ برای آستانه فاصله

همچنین با استفاده از تابع pairDistPlot(x, grp) در پکیج adegenet می‌توان به استخراج و ترسیم فواصل بین افراد دو گروه (یا جمعیت) مختلف پرداخت که در آن x، می‌تواند، فاصله، ماتریس یا اشیاء genind و DNAbin باشد و grp فاکتوری است که گروهی که افراد در آن عضو می‌باشند را نشان می‌دهد.

مثال: مجموعه داده H3N2 شامل ۱۹۰۳ فرد می‌باشد که به شش گروه (2001، 2002، 2003، 2004، 2005 و 2006) تقسیم شده‌اند :

```
> library(adegenet)

> data(H3N2)

> nInd(H3N2)

    [1] 1903


> other(H3N2)$epid
```

```
 [1] 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2002 2002 2002 2002 2001
[15] 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001 2001
.....
[1681] 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005
[1695] 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2005 2006
```

در اینجا به عنوان تمرین و برای نمایش بهتر فواصل بین افراد در گروه‌های مختلف، ۱۰۰ فرد را بطور تصادفی (از بین کل ۱۹۰۳ فرد) انتخاب می‌کنیم و تابع pairDistPlot را روی آنها اجرا می‌نماییم:

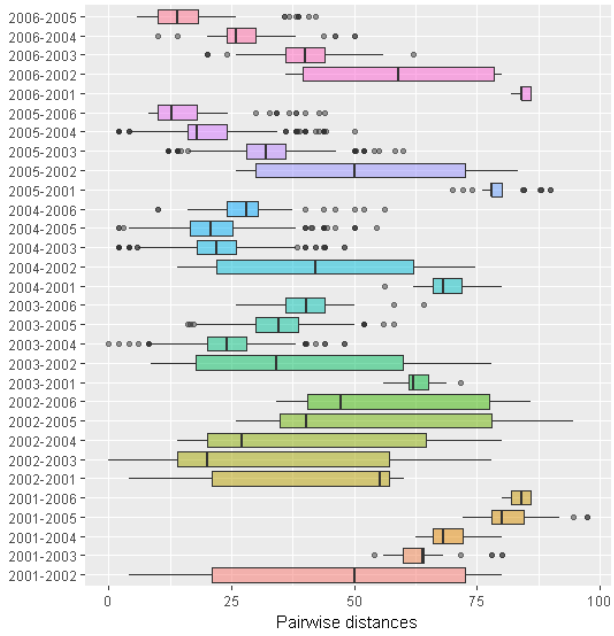
```
> set.seed(1)
> dat <- H3N2[sample(1:nInd(H3N2), 100)]
> temp <- pairDistPlot(dat, other(dat)$epid)
```

شیء ایجاد شده (temp) از نوع لیست است و از اجزاء مختلفی تشکیل شده است. جزء data شامل مقدار عددی فواصل جفتی بین افراد در گروه‌های مختلف می‌باشد و از اجزاء boxplot، violin و jitter می‌توان برای رسم نمودارهای مختلف (به ترتیب شکل‌های ۵-۳، ۵-۴ و ۵-۵) استفاده نمود:

```
> names(temp)
[1] "data"    "violin"  "boxplot" "jitter"
> class(temp)
[1] "list"
> names(temp)
[1] "data"    "violin"  "boxplot" "jitter"
> head(temp$data)
  distance    groups
1 22.00000 2003-2004
2 30.00000 2003-2005
3 17.20172 2003-2002
4 30.00000 2003-2005
```

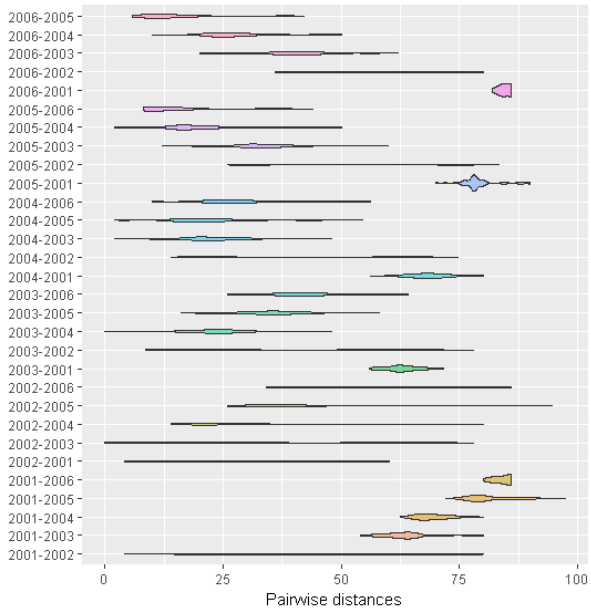
5	32.00000	2003-2005
6	32.00000	2003-2005

> temp$boxplot



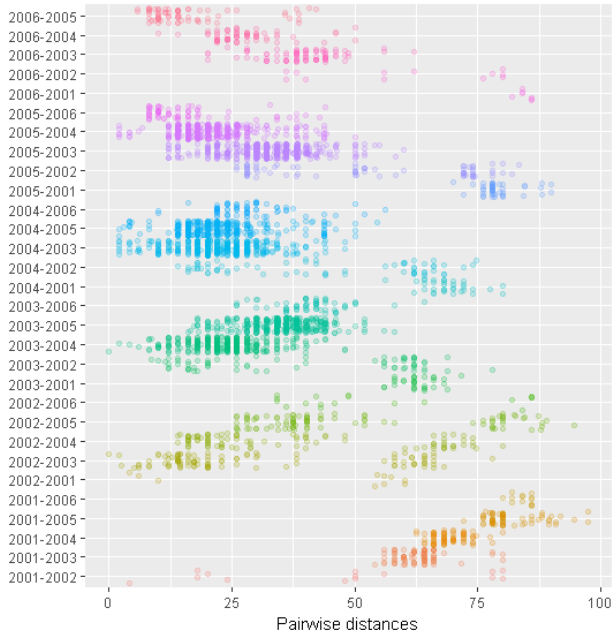
شکل ۵-۳– نمایش باکس‌پلات فواصل ژنتیکی بین جمعیت‌های مجموعه داده H3N2 برای ۱۰۰ فرد انتخاب شده تصادفی

> temp$violin



شکل ۵-۴– نمایش ویولون‌پلات فواصل ژنتیکی بین جمعیت‌های مجموعه داده H3N2 برای ۱۰۰ فرد انتخاب شده تصادفی

> temp$jitter



شکل ۵-۵– نمایش جیترپلات فواصل ژنتیکی بین جمعیت‌های مجموعه داده H3N2 برای ۱۰۰ فرد انتخاب شده تصادفی

ترسیم نقشه حرارتی و دندروگرام جمیعت‌ها

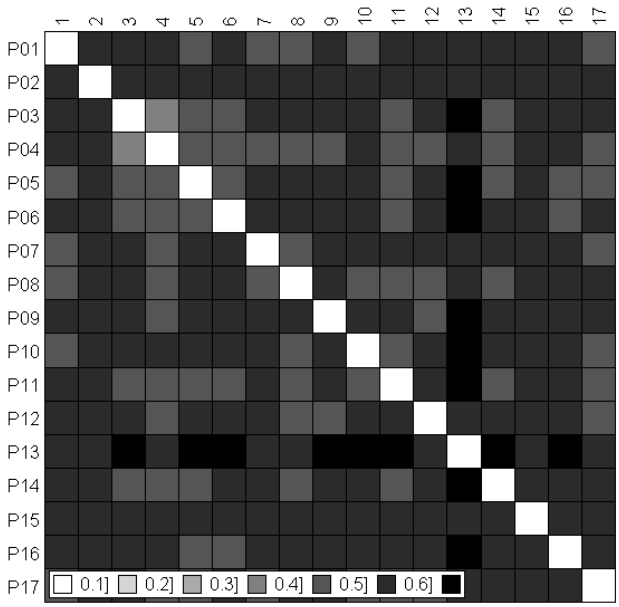
با استفاده از توابعی که در بخش‌های پیشین ذکر شد می‌توان فواصل ژنتیکی بین جمعیت‌ها را محاسبه و براساس آن نمودارهای نقشه حرارتی و دندروگرام جمیعت‌ها را ترسیم نمود. به عنوان مثال در اینجا فواصل ژنتیکی بین جمعیت‌ها در مجموعه داده nancycats محاسبه و ترسیم می‌شود:

```
> library(adegenet)
> data(nancycats)
> p.cats<-genind2genpop(nancycats)
> d2.p.cats<-dist.genpop(p.cats, met=2)
> d2m.p.cats<-as.matrix(d2.p.cats)
> d2m.p.cats [1:5,1:5]
```

	P01	P02	P03	P04	P05
P01	0.000000	0.595768	0.527762	0.524534	0.481428
P02	0.595768	0.000000	0.583150	0.549933	0.563584
P03	0.527762	0.583150	0.000000	0.374627	0.415319
P04	0.524534	0.549933	0.374627	0.000000	0.427732
P05	0.481428	0.563584	0.415319	0.427732	0.000000

فواصل فوق را می‌توان با استفاده از تابع table.paint بصورت نقشه حرارتی نمایش داد (شکل ۵-۶). تابع table.paint، از توابع پکیج ade4 می‌باشد که برای نمایش مقادیر یک آرایه به صورت طیفی از رنگ خاکستری بکار می‌رود.

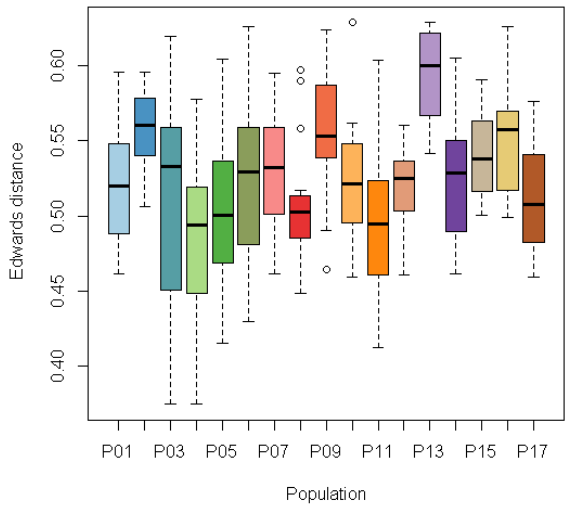
```
> table.paint(d2m.p.cats, col.labels=1:17)
```



شکل ۵-۶– نقشه حراراتی فواصل ژنتیکی بین جمعیت‌ها در مجموعه داده nancycats براساس معیار فاصله Edwards

همچنین می‌توان با استفاده از تابع boxplot، فاصله هر جمعیت از سایر جمعیت‌ها را با ترسیم باکس پلات نشان داد (شکل ۵-۷):

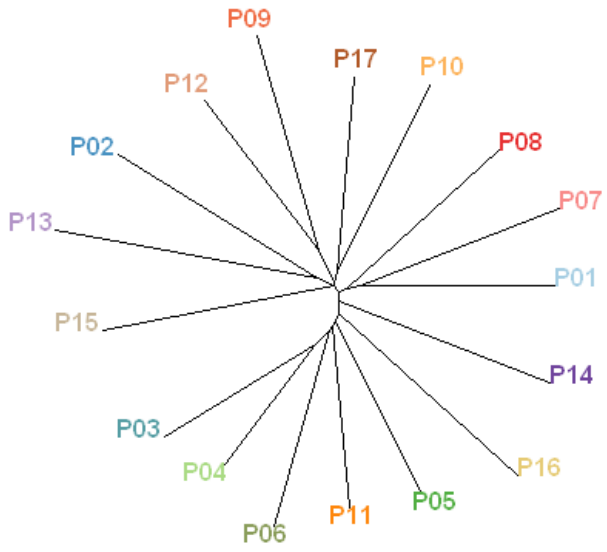
```
> temp <- d2m.p.cats
> diag(temp) <- NA
> boxplot(temp, col=funky(17), xlab="Population", ylab=" Edwards distance")
```



شکل ۵-۷– نمایش فواصل بین جمعیت‌های مجموعه داده nancycats مبتنی بر معیار فاصله Edwards

برای ترسیم دندورگرام مبتنی بر فواصل بین جمعیت‌ها از تابع nj در پکیج ape استفاده می‌کنیم. این تابع، به عنوان ورودی یک ماتریس فاصله یا یک شیء dist را می‌پذیرد و درخت فیلوژنی را به روش اتصال همسایه‌ها (Saitou and Nei, 1987) ترسیم می‌نماید. خروجی تابع nj یک شیء phylo است که توسط تابع plot بصورت نمودار نمایش داده می‌شود (شکل ۵-۸):

```
> library(ape)
> cats.tree<- nj(d2m.p.cats)
> cats.tree
        Phylogenetic tree with 17 tips and 15 internal nodes.
        Tip labels:
            P01, P02, P03, P04, P05, P06, ...

        Unrooted; includes branch lengths.
> class(cats.tree)
        [1] "phylo"

> plot(cats.tree, type="unr", tip.col=funky(17), font=2)
```



شکل ۵-۸– دندروگرام جمعیت‌های مجموعه داده nancycats مبتنی بر معیار فاصله Edwards

فصل ششم- آماره‌های افتراق جمعیت

انواع آماره‌های افتراق ژنتیکی

تنوع ژنی و افتراق ژنتیکی بین جمعیت‌ها، به عنوان ویژگی‌های بنیادین در ژنتیک جمعیت به شمار می‌روند و کاربردهای مهمی در علوم اکولوژی مولکولی، زیست‌شناسی تکاملی و حفاظت، انسان‌شناسی، ایمنی‌شناسی و غیره دارند. در ژنتیک جمعیت، تنوع ژنی معمولاً با هتروزیگوسیتی نشان داده می‌شود و افتراق ژنتیکی با معیارهایی مانند F_{ST} ، G_{ST} و سایر کمیت‌های مرتبط، برآورد می‌شود. مقادیر نزدیک به صفر برای این کمیت‌ها به معنی افتراق ناچیز و مقادیر نزدیک به یک، نشان‌دهنده افتراق کامل است. باید توجه داشت اگرچه F_{ST} به عنوان معیار افتراق ژنتیکی شناخته می‌شود، ولی مفهومی فراتر از یک کمیّت توصیفی دارد. آماره F_{ST} بطور مستقیم با واریانس فراوانی آللی بین جمعیت‌ها مرتبط می‌باشد. در نتیجه، F_{ST} میزان شباهت بین افراد در داخل جمعیت‌ها را نشان می‌دهد، بطوری که اگر مقدار F_{ST} کوچک باشد به این معنی است که فراوانی‌های آللی در داخل جمعیت‌های گوناگون، مشابه است و برعکس اگر مقدار F_{ST} بزرگ باشد به معنی تفاوتِ فراوانی‌های آللی بین جمعیت‌ها می‌باشد. اگر گزینش طبیعی در یکی از مکان‌های ژنی به نفع یکی از آلل‌ها عمل کند، مقدار F_{ST} در آن مکان ژنی بیشتر از سایر مکان‌های ژنی خواهد بود که تفاوت فراوانی آللی بین جمعیتی در آنها صرفاً ناشی از رانده شدن ژنتیکی می‌باشد. به همین ترتیب در پیمایش ژنومی، با محاسبه F_{ST} برای تک تک SNPها می‌توان مناطقی از ژنوم را شناسایی کرد که تحت گزینش تفرق‌زا[16] بوده‌اند. یکی از ایرادات وارد بر آماره F_{ST} اینست که حتی وقتی که میزان بالایی از افتراق ژنتیکی بین جمعیت‌ها وجود دارد مقدار آن کوچک است. در حقیقت مقادیر بزرگتر از ۰/۲۵ برای این آماره نشان‌دهنده افتراق ژنتیکی بالا در بین جمعیت‌ها می‌باشد. آمارهای دیگری، مشابه با F_{ST} نیز توسط مؤلفین گوناگون معرفی شده‌اند که هر کدام دارای مزایا و معایبی می‌باشند. آماره G_{ST} شبیه F_{ST} است و به عنوان معیار افتراق بین جمعیت‌ها بطور گسترده‌ای مورد استفاده قرار گرفته است. اما این آماره صرفاً هنگامی

[16] Diversifying selection

مناسب است که نقش رانده شدن ژنتیکی در تفاوت‌های بین جمعیتی مورد نظر نباشد. در نتیجه، زمینه مورد استفاده از آماره G_{ST} محدود است. همچنین فقط درصورتی که تنوع ژنی پایین باشد، G_{ST} آماره مناسبی خواهد بود. حتی در صورت پایین بودن تنوع هم، وابستگی G_{ST} به هتروزیگوسیتی درون جمعیت‌ها، سبب می‌شود که G_{ST} هنگام مقایسه جمعیت‌هایی که میانگین هتروزیگوسیتی متفاوت دارند، آماره مناسبی برای افتراق ژنتیکی نباشد. این وابستگی سبب می‌شود که مقایسه نشانگرهای مولکولی با نرخ موتاسیون متفاوت (نشانگرهای میکروستلایت در برابر آلوزایم‌ها یا DNA میتوکندریایی در برابر DNA کلروپلاستی) یا گونه‌های با اندازه جمعیت مؤثرِ متفاوت، دشوار شود. در این خصوص نویسندگان دیگری نیز آماره‌هایی معرفی نموده‌اند که مستقل از میزان تنوع ژنتیکی باشد، بطوریکه براساس آن‌ها بتوان مطالعات ژنتیکی که از نشانگرهای متفاوتی استفاده نموده‌اند را مورد مقایسه قرار داد که به عنوان نمونه می‌توان کمیّت افتراق ژنتیکی استاندارد شده (G'_{st}) را نام برد که توسط هدریک (Hedrick, 2005) ارائه گردید. با گسترش استفاده از نشانگرهای میکروستلایت که دارای تنوع بالایی می‌باشند، نقاط ضعف G_{ST} و آماره‌های مشابه آن، بیشتر آشکار شده است. درحقیقت برای مکان‌های ژنی با پلی‌مورفیسم بالا، G_{ST} قادر نیست هیچ‌گونه اطلاعات اضافه‌ای درخصوص افتراق ژنتیکی جمعیت‌ها ارائه دهد. جوست (Jost, 2008) به منظور حذف وابستگی کمیّت افتراق ژنتیکی به میانگین هتروزیگوسیتی داخل جمعیتی (Hs)، آماره D را معرفی و توصیه کرد که به عنوان معیار افتراق ژنتیکی، جایگزین G_{ST} شود، هر چند که در این موضوع اختلاف نظر وجود دارد. همچنین به منظور افزایش کارایی معیارهای افتراق ژنتیکیِ جمعیت در حالت‌های مختلف، آماره‌های دیگری توسعه یافته‌اند که به عنوان مثال می‌توان از آماره‌های F_{SR} و F_{RT} برای بررسی افتراق ژنتیکی در صورت وجود دسته‌بندی در جمعیت‌ها برحسب نواحی و یا N_{ST}، برای داده‌های توالی را نام برد. آماره‌های R_{ST} (برای داده‌های میکروستلایت) و Φ_{ST} (برای داده‌های توالی) فواصل جهشی بین آلل‌ها را منظور می‌نمایند و لذا در زمینه وسیع‌تری نسبت به آماره G_{ST} قابل مطالعه می‌باشند. همچنین از آماره Q_{ST} می‌توان در تجزیه صفات با تغییرات پیوسته استفاده نمود (Wright, 1921a؛ Wright, 1921b؛ Crow and Kimura, 1970؛ Nei, 1973؛ Nei, 1977؛ Nei and Chesser, 1983؛ Weir and Cockerham, 1984؛ Lynch and Crease, 1990. ؛Weir and Cockerham, 1984 ؛Hedrick, 1999 ؛Meirmans, 2006 ؛Akey *et al.*, 2002 ؛Culley *et al.*, 2002 ؛Goudet *et al.*, 1996 ؛Weir, 1996 ؛Mohammadi and Weir *et al.*, 2005 ؛Jost, 2008 ؛Holsinger, 2006 ؛Hedrick, 2005 ؛Ewens, 2004 ؛Prasanna, 2003 ؛Holsinger and Weir, 2009 ؛Foll and Gaggiotti, 2008 ؛Jost, 2008 ؛Hartl and Clark, 2007 ؛Ryman and Leimar, 2009 ؛Heller and Siegismund, 2009 ؛Jost, 2009 ؛ Guo *et al.*, 2009 ؛Dewar *et al.*, 2011 ؛Charlesworth and Charlesworth, 2010 ؛Gerlach *et al.*, 2010 ؛Whitlock, 2011 ؛ Meirmans and Hedrick, 2011 و Verity and Nichols, 2014).

برآورد آماره‌های افتراق ژنتیکی

از پکیج‌های مختلفی مانند poppr، hierfstat و mmod به منظور برآورد آماره‌های ژنتیکی پایه در جمعیت، می‌توان استفاده کرد. در اینجا به شرح توابع مربوطه و خروجی آن‌ها می‌پردازیم. یکی از توابع مفید در این زمینه، تابع poppr در پکیج poppr می‌باشد:

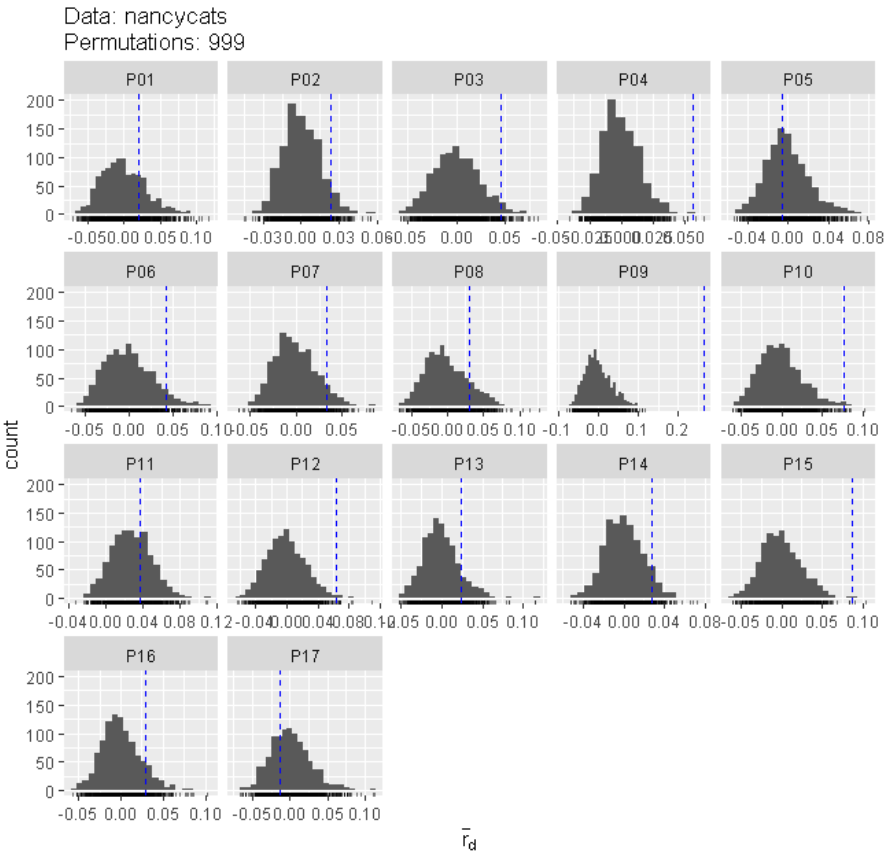
```
> library(adegenet)
> library(poppr)
> poppr(nancycats)
```

	Pop	N	MLG	eMLG	SE	H	G	lambda	E.5	Hexp	Ia	rbarD
1	P01	10	10	10	0.00E+00	2.3	10	0.9	1	0.649	0.1656	0.0211
2	P02	22	22	10	0.00E+00	3.09	22	0.955	1	0.701	0.1818	0.023
3	P03	12	12	10	0.00E+00	2.48	12	0.917	1	0.719	0.3546	0.0452
.....												
16	P16	12	12	10	0.00E+00	2.48	12	0.917	1	0.7	0.2345	0.0295
17	P17	13	13	10	7.30E-08	2.56	13	0.923	1	0.605	-0.0906	-0.0138
18	Total	237	237	10	0.00E+00	5.47	237	0.996	1	0.774	0.1721	0.0218

در نتایج فوق، آماره‌های SE، H، G، E.5، Hexp، Ia، و rbarD به ترتیب عبارت از خطای استاندارد تجزیه ترقیق، شاخص تنوع شانون-وینر، شاخص استودارت و تیلور، شاخص یکنواختی (Evenness)، تنوع ژنی نی (هتروزیگوسیتی مورد انتظار)، شاخص ارتباط و شاخص ارتباط استاندارد شده می‌باشند (Agapow and Burt, 2001؛ Brown *et al.*, 1980؛ Grünwald *et al.*, 2003؛ Haubold and Hudson, 2000؛ Heck *et al.*, 1975؛ Hurlbert, 1971؛ Nei, 1978؛ Ludwig and Reynolds, 1988؛ Simpson, 1949؛ Good, 1953؛ Lande, 1996؛ Oksanen *et al.*, 2012؛ Pielou, 1975؛ Shannon, 1948؛ Smith , *et al.*, 1993؛ و Stoddart and Taylor, 1988).

با استفاده از آرگومان sample در تابع poppr، می‌توان آمیزش تصادفی را شبیه سازی کرد به این ترتیب که ژنوتیپ‌ها در هر لوکوس، به تعداد دفعات مشخص شده در آرگومان sample، بُرمی‌خورند. با انتخاب گزینه TRUE برای آرگومان plot، نمودار فراوانی به تفکیک جمعیت‌ها رسم خواهد شد (شکل ۶-۱):

```
> poppr(nancycats, sample = 999, total = FALSE, plot = TRUE)
```



شکل ۶-۱- نتایج شبیه‌سازی آمیزش تصادفی به تفکیک جمعیت‌ها براساس ژنوتیپ‌های مشاهده شده در مجموعه داده nancycats

برای جزئیات بیشتر در مورد محاسبه آماره‌های ژنتیکی در آرگومان‌های تابع poppr و خروجی آن، دستور زیر را اجرا و به صفحه مرورگری که باز خواهد شد، مراجعه نمایید:

```
> help(poppr)
```

با استفاده از تابع locus_table از پکیج poppr، می‌توان آماره‌های ژنتیکی را به تفکیک لوکوس‌ها، محاسبه نمود:

```
locus_table(x, index = "simpson", lev = "allele")
```

در تابع locus_table، آرگومان index، شاخص تنوع را تعیین می‌کند که بصورت پیش‌فرض، simpson (برای محاسبه شاخص سیمپسون) است ولی می‌توان آن را shannon (برای محاسبه شاخص شانون-وینر) یا invsimpson (برای محاسبه عکسِ شاخص سیمپسون که به عنوان شاخص استودارت و تیلور نیز شناخته

می‌شود) قرار داد. با استفاده از آرگومان lev، می‌توان تعیین کرد که تجزیه تنوع در سطح آلل (allele) یا ژنوتیپ (genotype) انجام شود.

به عنوان مثال آماره‌های ژنتیکی به تفکیک لوکوس‌ها برای جمعیت نهم از مجموعه داده nancycats، با استفاده از تابع locus_table بصورت زیر محاسبه می‌شوند:

```
> locus_table(nancycats[pop = 9])
```

allele = Number of observed alleles

1-D = Simpson index

Hexp = Nei's 1978 gene diversity

---

summary

locus	allele	1-D	Hexp	Evenness
fca8	4.00	0.66	0.70	0.86
fca23	6.00	0.72	0.76	0.75
fca43	3.00	0.54	0.57	0.86
fca45	8.00	0.82	0.87	0.83
fca77	5.00	0.65	0.69	0.74
fca78	3.00	0.54	0.57	0.86
fca90	5.00	0.62	0.66	0.69
fca96	5.00	0.73	0.77	0.85
fca37	4.00	0.62	0.66	0.80
mean	4.78	0.66	0.69	0.80

تابع basic.stats در پکیج hierfstat نیز برآوردهای مفیدی از آماره‌های ژنتیکی پایه ارائه می‌کند. به عنوان مثال محاسبه آماره‌های ژنتیکی پایه در مجموعه داده nancycats با استفاده از تابع basic.stats به شرح زیر می‌باشد:

```
> library(hierfstat)
> basic.stats(nancycats)
```

$perloc

	Ho	Hs	Ht	Dst	Htp	Dstp	Fst	Fstp	Fis	Dest
fca8	0.667	0.779	0.8619	0.0829	0.8671	0.0881	0.0962	0.1016	0.1438	0.3987
fca23	0.6838	0.7439	0.7994	0.0555	0.8029	0.0589	0.0694	0.0734	0.0809	0.2302
fca43	0.6814	0.7442	0.7937	0.0495	0.7968	0.0526	0.0623	0.066	0.0844	0.2054
fca45	0.71	0.7085	0.7642	0.0557	0.7679	0.0594	0.0729	0.0774	-0.0021	0.2039

fca77	0.6295	0.7828	0.8659	0.0831	0.8711	0.0883	0.096	0.1014	0.1958	0.4067
fca78	0.5773	0.6339	0.6773	0.0434	0.6801	0.0462	0.0641	0.0679	0.0893	0.1261
fca90	0.6454	0.7408	0.8144	0.0736	0.819	0.0782	0.0904	0.0955	0.1287	0.3017
fca96	0.6259	0.6747	0.7657	0.091	0.7714	0.0967	0.1189	0.1254	0.0723	0.2973
fca37	0.4485	0.5671	0.6027	0.0356	0.6049	0.0379	0.0591	0.0626	0.2091	0.0874

$overall

Ho	Hs	Ht	Dst	Htp	Dstp	Fst	Fstp	Fis	Dest
0.6299	0.7083	0.7717	0.0634	0.7757	0.0674	0.0821	0.0869	0.1108	0.231

همانطور که مشاهده می‌شود تابع basic.stats آماره‌های ژنتیکی پایه را علاوه بر کل جمعیت، به تفکیک لوکوس‌ها نیز محاسبه و ارائه می‌نماید که به شرح ذیل می‌باشند:

Ho: میانگین هتروزیگوسیتی مشاهده شده

Hs: میانگین تنوع ژنتیکی درون جمعیتی، توجه کنید که گاهی اوقات این آماره، به اشتباه هتروزیگوسیتی مورد انتظار پنداشته می‌شود.

Ht: تنوع ژنی کل

Dst: تنوع ژنی بین نمونه‌ها، $Dst = Ht - Hs$

Htp: Ht تصحیح شده (Ht')، $Ht' = Hs - Dst'$

Dstp: Dst تصحیح شده (Dst')

Fst: معیار افتراق جمعیت بدلیل ساختار ژنتیکی، $Fst = Dst/Ht$

توجه داشته باشید که آماره F_{ST} متفاوت از آماره G_{ST} است که توسط نی (Nei, 1973) ارائه شد، می‌باشد. آماره G_{ST} ارائه شده توسط نی، برآوردی از F_{ST} صرفاً براساس فراوانی‌های آللی می‌باشد.

Fstp: F_{ST} تصحیح شده (F_{ST}')، $Fst' = Dst'/Ht'$

Fis: ضریب درون زادآوری (معیار انحراف از تعادل هاردی واینبرگ در زیرجمعیت‌ها)، $Fis = 1 - Ho/Hs$

Dest: معیار افتراق جمعیت طبق تعریف جوست (Jost, 2008)

تابع مفید دیگر برای محاسبه آماره‌های ژنتیکی پایه در جمعیت، تابع diff_stats از پکیج mmod می‌باشد. در اینجا کاربرد این تابع را بر روی مجموعه داده microbov شرح می‌دهیم:

```
> library(mmod)
> data(microbov)
> mic.stat<-diff_stats(microbov)
> mic.stat
$per.locus
```

	Hs	Ht	Gst	Gprime_st	D
INRA63	0.564	0.7063	0.2015	0.4881151	0.3497
INRA5	0.5756	0.6037	0.0465	0.1170902	0.0709
ETH225	0.7198	0.783	0.0808	0.3070142	0.2418
ILSTS5	0.4638	0.5425	0.1451	0.2868814	0.1573
HEL5	0.693	0.793	0.1261	0.4362548	0.3491
HEL1	0.6224	0.762	0.1831	0.5128954	0.3959
INRA35	0.4544	0.4932	0.0788	0.1538388	0.0763
ETH152	0.6119	0.6845	0.1061	0.2906344	0.2004

……..

$global

Hs	Ht	Gst_est	Gprime_st	D_het	D_mean
0.6545	0.7456	0.1222	0.3756	0.282512	0.2347

همانطور که مشاهده می‌شود در اینجا نیز آماره‌های ژنتیکی پایه علاوه بر کل جمعیت، به تفکیک لوکوس‌ها هم محاسبه و ارائه شده‌اند. در نتایج فوق آماره‌های ارائه شده عبارت از توسط G_{ST} نی (Nei, 1973)، Gprime_st یا G_{ST}' هدریک (Hedrick, 2005) و D جوست (Jost, 2008) می‌باشند. بعلاوه با استفاده از آرگومان phi_st = TRUE در تابع diff_stats (که در حالت پیش فرض phi_st = FALSE می‌باشد) می‌توان آماره Phi'st را که نسخه تصحیح شده Phi'st (Meirmans, 2005) است، محاسبه نمود.

توجه داشته باشید که هر یک از آماره‌های تابع diff_stats، خود بصورت تابع مستقلی قابل محاسبه می‌باشند. به عنوان مثال دستورات زیر را اجرا و نتایج را مشاهده نمایید:

```
> D_Jost(microbov)
> Gst_Hedrick(microbov)
> Gst_Nei(microbov)
> Phi_st_Meirmans(microbov)
```

ترسیم نمودار برای آماره‌های افتراق ژنتیکی

با استفاده از توابع موجود در پکیج‌های ggplot2 و reshape2 و براساس نتایج ایجاد شده توسط تابع diff_stats، می‌توان نموداری از آماره‌های ژنتیکیِ محاسبه شده، ترسیم و آنها را مورد مقایسه قرار داد (مطالب این بخش در ادامه بخش قبل می‌باشد و لذا اگر دستورات بخش قبل را اجر نکرده باشید ممکن است با پیام خطا مواجه شوید، همچنین توجه کنید که بایستی قبل از اجرای دستورات زیر پکیج‌های ggplot2 و reshape2 را نصب کرده باشید):

```
>  library(ggplot2)
> library(reshape2)
> per.locus<- melt(mic.stat$per.locus, varnames =c("Locus", "Statistic"))
> per.locus
```

	Locus	Statistic	value
1	INRA63	Hs	0.564
2	INRA5	Hs	0.5756
3	ETH225	Hs	0.7198
4	ILSTS5	Hs	0.4638
5	HEL5	Hs	0.693
6	HEL1	Hs	0.6224
7	INRA35	Hs	0.4544
8	ETH152	Hs	0.6119

……..

melt، یکی از توابع زیرمجموعه پکیج reshape2 است. با استفاده از این تابع می‌توان ستون‌های یک ماتریس را زیر هم آورد. دستور فوق اطلاعات ستون‌های جزء per.locus از شیء mic.stat که یک ماتریس ۳۰×۵ است را، در زیر هم قرار می‌دهد:

```
> class(mic.stat$per.locus)
     [1] "matrix"
> dim(mic.stat$per.locus)
     [1] 30  5
```

در مرحله بعد با استفاده از آماره‌هایی که توسط تابع diff_stats بر روی کل جمعیت محاسبه شد، یک قالب داده، تشکیل می‌دهیم:

```
> stats<- c("Hs", "Ht", "Gst", "Gprime_st", "D", "D")
> glob<- data.frame(Statistic = stats, value = mic.stat$global)
```

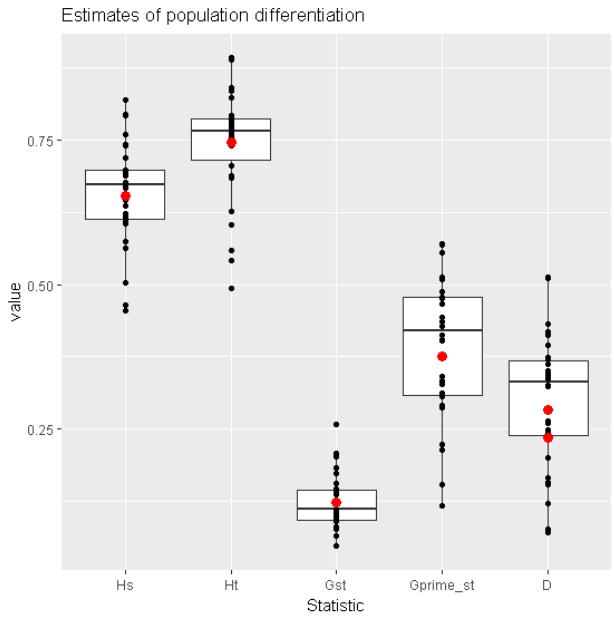
> glob

	Statistic	value
Hs	Hs	0.6545
Ht	Ht	0.7456
Gst_est	Gst	0.1222
Gprime_st	Gprime_st	0.3756
D_het	D	0.2825
D_mean	D	0.2347

اکنون با استفاده از تابع ggplot می‌توان برای هر یک از آماره‌های محاسبه شده، یک باکس پلات به شرح زیر ترسیم نمود (شکل ۶-۲):

```
> ggplot(per.locus, aes(x = Statistic, y = value)) + geom_boxplot() + geom_point() +
geom_point(size =rel(3), color ="red", data = glob) + ggtitle("Estimates of population differentiation")
```

این نمودار امکان مقایسه آماره‌های ژنتیکی افتراق جمیعت و هم توزیع هر یک از آنها در لوکوس‌ها را، نشان می‌دهد. همانطور که در نمودار مشخص است، مقدار آماره G_{ST} چه در جمعیت اصلی و چه در زیرجمعیت‌ها، کمتر از آماره‌های G_{ST}' و D می‌باشد. این نتیجه توسط محققان دیگر نیز گزارش شده است (Heller and Siegismund, 2009).



شکل ۶-۲- مقایسه آماره‌های افتراق ژنتیکی در مجموعه داده microbov. نقاط سیاه نشان‌دهنده‌ی مقدار آماره در لوکوس‌های مختلف و نقطه قرمز در هر باکس‌پلات، نشان‌دهنده‌ی مقدار محاسبه شده آماره در جمعیت است.

برآورد اختصاصی و آزمون آماره F_{ST}

از توابع fstat و Fst به ترتیب در پکیج‌های hierfstat و pegas، می‌توان برای محاسبه آماره‌های F، استفاده نمود. به عنوان مثال محاسبه آماره‌های F در مجموعه داده nancycats با استفاده از این توابع به شرح زیر می‌باشد:

```
> library(hierfstat)
> library(adegenet)
>  data(nancycats)
> fstat(nancycats)
```

	pop	Ind
Total	0.0849496	0.1953
Pop	0	0.1206

نتایج فوق از محاسبه بر روی کل جمعیت به دست آمده است و در آن، F_{ST}=0.0849496، F_{IT}=0.1953 و F_{IS}=0.1206 می‌باشد. برای محاسبه آماره‌های F به تفکیک لوکوس‌ها، از تابع Fst در پکیج pegas استفاده می‌کنیم:

```
> library(pegas)
> f.cats<-Fst(as.loci(nancycats))
> f.cats
```

	Fit	Fst	Fis
fca8	0.244742	0.1015	0.1595
fca23	0.1646295	0.0675	0.1042
fca43	0.1514487	0.0689	0.0886
fca45	0.1010807	0.0979	0.0035
fca77	0.2790495	0.1004	0.1986
fca78	0.184249	0.0703	0.1226
fca90	0.2098744	0.0917	0.1301
fca96	0.2034755	0.1074	0.1076
fca37	0.2604033	0.0699	0.2049

براساس نتایج فوق میزان F_{ST}، در لوکوس fca96 بیشتر از سایر لوکوس‌ها می‌باشد.

نکته: دلیل استفاده از as.loci، در تابع Fst، اینست که این تابع بر روی اشیاء loci عمل می‌کند و تابع as.loci دادهای nancycats که از نوع شیء genind است را به شیء loci تبدیل می‌کند، تا تابع Fst روی آن قابل اعمال باشد.

ترسیم نقشه حرارتی ودندروگرام مبتنی بر F_{ST}

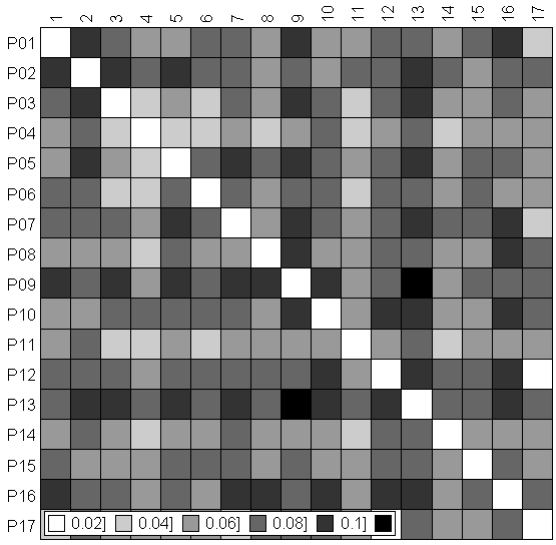
با استفاده از تابع pairwise.fst در پکیج hierfstat می‌توان آماره F_{ST} جفتی نی (Nei, 1973) را بصورت دو به دو، بین تمام جمعیت‌ها محاسبه نمود. آماره F_{ST} جفتی به روش زیر در مجموعه داده nancycats قابل محاسبه می‌باشد:

```
> library(adegenet)
> library(hierfstat)
> p.fst.cats<- pairwise.fst(nancycats, res.type="matrix")
> class(p.fst.cats)
        [1] "matrix"
> dim(p.fst.cats)
        [1] 17 17
> head(p.fst.cats)[1:5,1:5]
```

	P01	P02	P03	P04	P05
P01	0.000000	0.080185	0.071408	0.049925	0.059069
P02	0.080185	0.000000	0.082009	0.069855	0.082953
P03	0.071408	0.082009	0.000000	0.025716	0.053047
P04	0.049925	0.069855	0.025716	0.000000	0.038335
P05	0.059069	0.082953	0.053047	0.038335	0.000000

با استفاده از آرگومان res.type می‌توان خروجی تابع pairwise.fst را بصورت ماتریس (res.type="matrix") یا شیئی از نوع فاصله (res.type="dist") تنظیم نمود. لذا آماره F_{ST} جفتی نی (Nei, 1973) معیاری از فاصله بین جمعیت‌ها می‌باشد. این فواصل را می‌توان با استفاده از تابع table.paint بصورت نقشه حرارتی نمایش داد (شکل ۶-۳). تابع table.paint، از توابع پکیج ade4 می‌باشد که برای نمایش مقادیر یک آرایه بصورت طیفی از رنگ خاکستری بکار می‌رود.

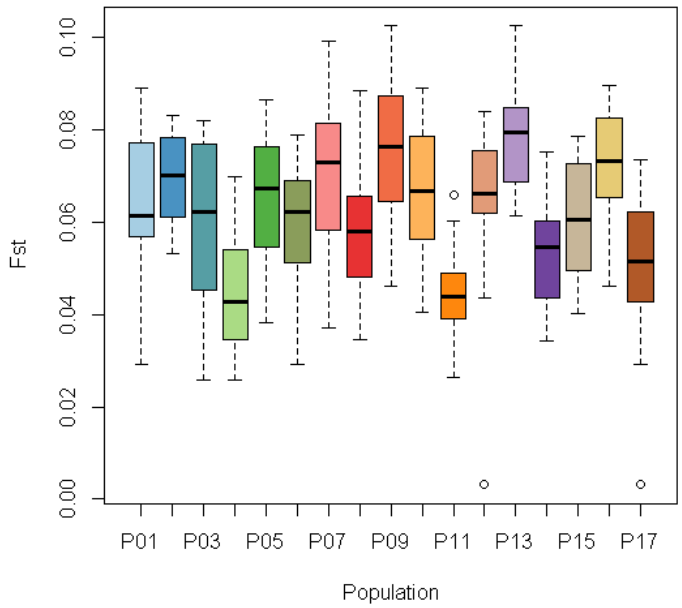
```
> table.paint(p.fst.cats, col.labels=1:17)
```



شکل ۶-۳– نقشه حراراتی فواصل ژنتیکی بین جمعیت‌ها در مجموعه داده nancycats براساس مقادیر F_{ST} جفتی

همچنین می‌توان از ماتریس F_{ST}های جفت جمعیت‌ها برای ترسیم باکس پلات مربوط به فواصل هر جمعیت از سایر جمعیت‌ها، استفاده نمود (شکل ۶-۴):

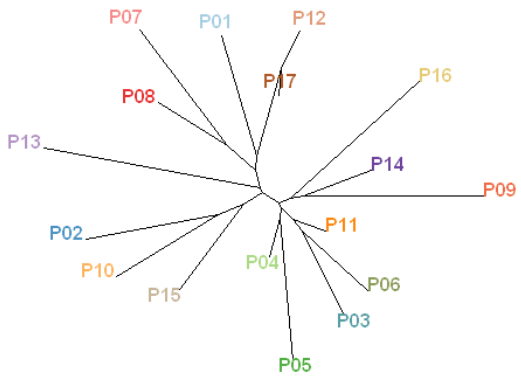
```
> temp <- p.fst.cats
> diag(temp) <- NA
> boxplot(temp, col=funky(17), xlab="Population", ylab="Fst")
```



شکل ۶-۴- نمایش فواصل بین جمعیت‌های مجموعه داده nancycats مبتنی بر مقادیر F_{ST} جفتی

همانطور که اشاره شد مقادیر آماره F_{ST} دو به دو بین جمعیت‌ها معیاری از فاصله بین آن‌ها می‌باشد. لذا می‌توان براساس این مقادیر، دندروگرام جمعیت‌ها را رسم نمود. برای اینکار از تابع nj در پکیج ape استفاده می‌کنیم. این تابع، به عنوان ورودی یک ماتریس فاصله یا یک شیء dist را می‌پذیرد و برآوردهای ترسیم درخت به روش اتصال همسایه‌ها (Saitou and Nei, 1987) را انجام می‌دهد. خروجی تابع nj یک شیء phylo است که توسط تابع plot بصورت نمودار نمایش داده می‌شود (شکل ۶-۵):

```
> library(ape)
> cats.tree<- nj(p.fst.cats)
> cats.tree
> plot(cats.tree, type="unr", tip.col=funky(17), font=2)
```



شکل ۶-۵- دندروگرام جمعیت‌های مجموعه داده nancycats مبتنی بر مقادیر F_{ST}

در تابع فوق، آرگومان type="unr" به نوع درخت بدون ریشه (unrooted) اشاره دارد و آرگومان tip.col رنگ سرشاخه‌های درخت که همان اسامی جمعیت‌ها است را، مشخص می‌کند.

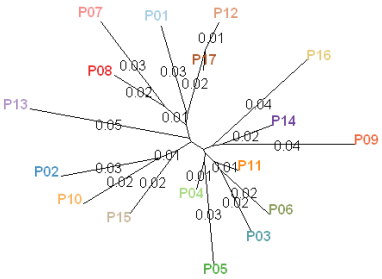
برای افزودن طول شاخه‌ها به درخت، می‌توان از تابع edgelabels استفاده کرد. بدین منظور از اطلاعات موجود در شیء phylo استفاده می‌شود. همانطور که اشاره شد خروجیِ تابع nj، از این نوع می‌باشد. جزء edge.length از شیء phylo، طول شاخه را نشان می‌دهد.

```
> class(cats.tree)
[1] "phylo"
```

```
> str(cats.tree)
        List of 4
         $ edge      : int [1:31, 1:2] 18 27 31 31 27 18 20 21 23 23 ...
         $ edge.length: num [1:31] 0.00457 0.00587 0.02816 0.02498 0.02246 ...
         $ tip.label  : chr [1:17] "P01" "P02" "P03" "P04" ...
         $ Nnode      : int 15
         - attr(*, "class")= chr "phylo"
         - attr(*, "order")= chr "cladewise"
> cats.tree$edge.length
          [1]  0.004565382  0.005873919  0.028157656  0.024979348  0.022463137
          [6]  0.004095793  0.001076974  0.002267455  0.008422905  0.029912349
         ……….
         [26] -0.005599320  0.026304080  0.007775016  0.030475126  0.017100777
         [31]  0.046197528


> len<- round(cats.tree$edge.length,2)
> len
          [1]  0.00  0.01  0.03  0.02  0.02  0.00  0.00  0.00  0.01  0.03  0.00  0.00
         [13]  0.02  0.02  0.01  0.00  0.00  0.04  0.02  0.04  0.00  0.00  0.00  0.02
         [25]  0.01 -0.01  0.03  0.01  0.03  0.02  0.05
```

همانطور که مشاهده می‌شود برخی از مقادیر، صفر است. با استفاده از کروشه در دستور len[len>0]، مقادیر غیرصفر مربوط به طول شاخه را متمایز می‌کنیم. همچنین با استفاده از تابع which، ترتیب قرار گرفتنِ مقادیر غیرصفر در وکتور len را، مشخص می‌نماییم و براساس این مقادیر، اعداد مربوط به طول شاخه‌ها را توسط تابع edgelabels، به درخت اضافه می‌نماییم (شکل ۶-۶).

```
> len.non<-len[len>0]
> len.non
          [1] 0.01 0.03 0.02 0.02 0.01 0.03 0.02 0.02 0.01 0.04 0.02 0.04 0.02 0.01 0.03
         [16] 0.01 0.03 0.02 0.05
> ord.len<-which(len>0)
> ord.len
          [1]  2  3  4  5  9 10 13 14 15 18 19 20 24 25 27 28 29 30 31
> edgelabels(len.non, ord.len, frame="n")
```



شکل ۶-۶- دندروگرام جمعیت‌های مجموعه داده nancycats مبتنی بر مقادیر Fst جفتی با درج طول یال‌ها

ضریب درون‌زادآوری افراد

ضریب درون‌زادآوری[17] (F) برای یک فرد بصورت احتمال به ارث رسیدن دو آلل مشابه از یک جد مشترک تعریف می‌شود. این کمیّت برای نخستین بار توسط رایت (Wright, 1922) معرفی شد اما اخیراً روش‌های جدید برآورد یا جنبه‌های مختلف آن مجدداً مورد توجه قرار گرفته است (Vogl *et al.*, 2002؛ Chapuis and ؛Anderson and Weir, 2007 ؛Carothers *et al.*, 2006 ؛Leutenegger *et al.*, 2003 ؛Estoup, 2007 ؛Chybicki and Burczyk, 2009 ؛Wang, 2011a ؛ Wang, 2011b ؛Hall *et al.*, 2012). اگر فراوانی آلل iاُم در جمعیت را بصورت p_i در نظر بگیریم و h متغیری باشد که مقدار آن درصورت هموزیگوت بودن فرد برابر با یک و درغیراینصورت مساوی با صفر باشد، آنگاه احتمال هموزیگوت بودن یک لوکوس عبارت خواهد بود از:

$$F + (1 - F) \sum p_i^2$$

و احتمال هتروزیگوت بودن آن لوکوس عبارتست از:

$$1 - (F + (1 - F) \sum p_i^2)$$

در اینحالت میزان درستنمایی عبارت از احتمال رخداد مشاهده شده (وضعیت هموزیگوت یا هتروزیگوت) است. در صورت وجود چند لوکوس، این میزان درستنمایی برای همه آنها جمع می‌شود.

تابع inbreeding در پکیج adgenet ضریب درون‌زادآوری افراد (F) را با استفاده از توابع درستنمایی محاسبه می‌نماید. خروجی این تابع، چگالی احتمال F یا نمونه‌ای از مقادیر F از این توزیع می‌باشد. این تابع بر روی افراد شیء genind با هر سطحی از پلوئیدی بزرگتر از یک، قابل اعمال است. در اینجا به عنوان مثال نحوه کار با تابع inbreeding بر روی مجموعه داده microbov شرح داده می‌شود:

```
> library(adegenet)
> data(microbov)
> nPop(microbov)
[1] 15
```

همانطور که مشاهده می‌شود مجموعه داده microbov از ۱۵ جمعیت مختلف تشکیل شده است. در اینجا قصد داریم ضریب درون‌زادآوری را برای افراد منتسب به جمعیت Salers، محاسبه نماییم، لذا با استفاده از تابع seppop افرادِ این جمعیت را در شیء جداگانه (sal) ذخیره می‌کنیم:

[17] Inbreeding

```
> levels(pop(microbov))
 [1] "Borgou"          "Zebu"            "Lagunaire"       "NDama"
 [5] "Somba"           "Aubrac"          "Bazadais"        "BlondeAquitaine"
 [9] "BretPieNoire"    "Charolais"       "Gascon"          "Limousin"
[13] "MaineAnjou"      "Montbeliard"     "Salers"
> sal<- seppop(microbov)$Salers
> sal
/// GENIND OBJECT /////////
 // 50 individuals; 30 loci; 373 alleles; size: 114.4 Kb
 // Basic content
   @tab:  50 x 373 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 5-22)
   @loc.fac: locus factor for the 373 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 2-2)
   @type:  codom
   @call: .local(x = x, i = i, j = j, treatOther = ..1, quiet = ..2, drop = drop)
 // Optional content
   @pop: population of each individual (group size range: 50-50)
   @other: a list containing: coun  breed  spe
```

همانطور که مشخص است شیء ایجاد شده از نوع genind است لذا تابع inbreeding بر روی آن قابل اعمال می‌باشد. نحوه عمل تابع inbreeding بدین ترتیب است که برای هر فرد، یک توزیع احتمال از مقادیر F ایجاد می‌نماید. با استفاده از آرگومان N در تابع inbreeding مشخص می‌کنیم که چند مقدار از این توزیع نمونه‌برداری شود. خروجی تابع inbreeding لیستی است که اجزاء آن به نام افرادی است که در شیء ورودی تابع inbreeding (یعنی sal) وجود دارند. به عبارت دیگر با توجه به اینکه شیء sal از ۵۰ فرد تشکیل شده بود، خروجی تابع inbreeding متشکل از لیستی با ۵۰ جزء خواهد بود. هر یک از این اجزاء حاوی مقادیر نمونه‌برداری شده (۱۰۰ نمونه) از توزیع احتمال F (ضریب درون‌زادآوری) برای فرد مربوطه می‌باشند:

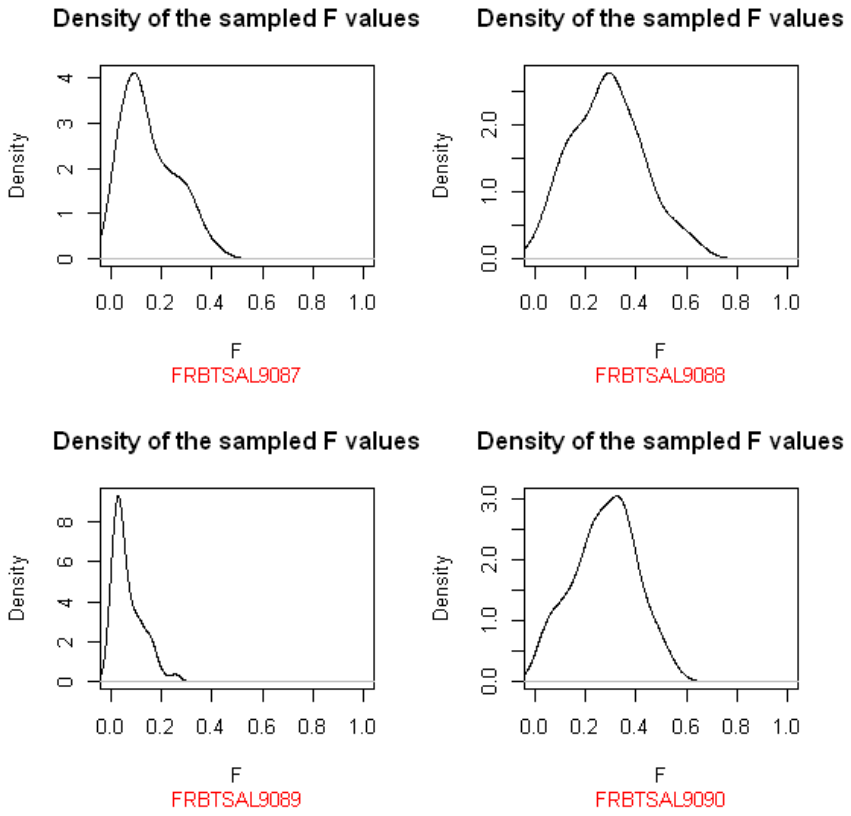
```
> Fsamp <- inbreeding(sal, N=100)
```

```
> class(Fsamp)
        [1] "list"
> names(Fsamp)
         [1] "FRBTSAL9087" "FRBTSAL9088" "FRBTSAL9089" "FRBTSAL9090" "FRBTSAL9091"
         [6] "FRBTSAL9093" "FRBTSAL9094" "FRBTSAL9095" "FRBTSAL9096" "FRBTSAL9097"
        ……..
        [41] "FRBTSAL9271" "FRBTSAL9272" "FRBTSAL9273" "FRBTSAL9275" "FRBTSAL9276"
        [46] "FRBTSAL9277" "FRBTSAL9280" "FRBTSAL9283" "FRBTSAL9284" "FRBTSAL9285"
> length(names(Fsamp))
        [1] 50
> Fsamp[[1]]
                 [1] 0.078264720 0.259675076 0.075103836 0.020989780 0.116340389 0.074342318
                 [7] 0.039481469 0.064311344 0.067772607 0.315347244 0.094071721 0.219228833
                ……..
                [91] 0.052821772 0.048722990 0.113874606 0.279483366 0.233993069 0.378095354
                [97] 0.128034514 0.149005765 0.398059489 0.020428177
> Fsamp[[50]]
                 [1] 0.161531544 0.046844345 0.093907102 0.105135624 0.281758417 0.017387011
                 [7] 0.222762652 0.048593849 0.063405172 0.097783840 0.081777690 0.174822101
                ……
                [91] 0.058460423 0.177757659 0.056394809 0.209374032 0.003735147 0.138915739
                [97] 0.158360416 0.091668892 0.431475711 0.206652345
```

با استفاده از دستورات زیر می‌توان منحنی توزیع احتمال F (ضریب درون‌زآدآوری) برای چهار فرد اول از جمعیت Salers (یعنی FRBTSAL9087، FRBTSAL9088، FRBTSAL9089 و FRBTSAL9090) را در یک پنجره مشاهده نمود (شکل ۶-۷):

```
> opar <- par()
> par(mfrow=c(2,2))
> invisible(sapply(Fsamp[1], function(e) plot(density(e), xlab="F", xlim=c(0,1), main="Density of the sampled F values", sub= names(Fsamp)[1], col.sub="red")))
```

```
> invisible(sapply(Fsamp[2], function(e) plot(density(e), xlab="F", xlim=c(0,1), main="Density of the sampled F values", sub= names(Fsamp)[2], col.sub="red")))

> invisible(sapply(Fsamp[3], function(e) plot(density(e), xlab="F", xlim=c(0,1), main="Density of the sampled F values", sub= names(Fsamp)[3], col.sub="red")))

> invisible(sapply(Fsamp[4], function(e) plot(density(e), xlab="F", xlim=c(0,1), main="Density of the sampled F values", sub= names(Fsamp)[4], col.sub="red")))

> par(opar)
```



شکل ۶-۷- منحنی توزیع احتمال ضریب درون‌زادآوری (F) برای چهار فرد اول از جمعیت Salers در مجموعه داده microbov

برآورد مقدار F را برای هر فرد می‌توان از میانگین مقادیر نمونه برداری شده از توزیع احتمال F مربوط به آن فرد بدست آورد. بنابراین با اعمال تابع میانگین بر روی لیست Fsamp، مقادیر برآورد شده برای ۵۰ فرد در جمعیت Salers بدست می‌آید:

```
> Fmean=sapply(Fsamp, mean)

> Fmean
```

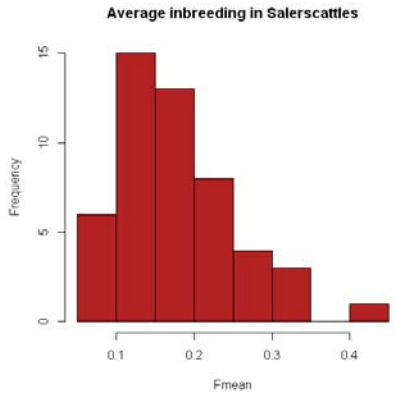
FRBTSAL9087	FRBTSAL9088	FRBTSAL9089	FRBTSAL9090	FRBTSAL9091	FRBTSAL9093
0.15332179	0.28975814	0.06812344	0.27792897	0.21113451	0.15134634
FRBTSAL9094	FRBTSAL9095	FRBTSAL9096	FRBTSAL9097	FRBTSAL9099	FRBTSAL9100
0.14064447	0.13523498	0.23867514	0.14517788	0.23901623	0.11081359

…….....

FRBTSAL9273	FRBTSAL9275	FRBTSAL9276	FRBTSAL9277	FRBTSAL9280	FRBTSAL9283
0.33374135	0.20289577	0.25636339	0.31404355	0.15840179	0.18645988
FRBTSAL9284	FRBTSAL9285				
0.10442393	0.11805946				

با رسم نمودار فراوانی تصویر بهتری از توزیع ضریب درون‌زادآوری در جمعیت بدست می‌آید (شکل ۶-۸):

```
> hist(Fmean, col="firebrick", main="Average inbreeding in Salerscattles")
```



شکل ۶-۸ - مقادیر برآورد شده ضریب درون‌زادآوری برای افراد جمعیت Salers در مجموعه داده microbov

با استفاده از تابع which می‌توان افرادی که واجد ضریب درون‌زادآوری بیشتر از یک حد آستانه (مثلاً ۰/۴) می‌باشند را متمایز نمود:

```
> which(Fmean>0.4)
FRBTSAL9266
         37
```

فصل هفتم- تجزیه چندمتغیره داده‌های ژنتیکی

تجزیه به مؤلفه‌های اصلی (PCA)

تجزیه به مؤلفه‌های اصلی در زمره قدیمی‌ترین و پرکاربردترین تکنیک‌های چند متغیره می‌باشد. این روش تجزیه برای نخستین بار توسط پیرسون (Pearson, 1901) و همچنین به طور مستقل توسط هتلینگ (Hotelling, 1933) معرفی شد و ایده اصلی آن توصیف تغییرات موجود در داده‌های چند متغیره در قالب مجموعه‌ای از متغیرهای غیرهمبسته است که خودِ این متغیرها، حاصل از ترکیبی خطی از متغیرهای اولیه می‌باشند. لذا تجزیه به مؤلفه‌های اصلی تکنیکی آماری است که به منظور خلاصه‌سازی و کاهش ابعاد داده‌ها استفاده می‌شود. با عمل کاهش ابعاد داده‌ها می‌توان تصویر واضح‌تری از گروه‌بندی افراد و ارتباط آنها با یکدیگر ارائه داد. فرض کنید تعداد زیادی متغیر را در افرادی اندازه‌گیری کرده‌ایم. براساس هر یک از این متغیرها روابط افراد با یکدیگر متفاوت خواهد بود. به عنوان مثال براساس یک متغیر، دو فرد ممکن است به هم نزدیک باشند، ولی براساس متغیر دیگر از هم فاصله داشته باشند. حال اگر بتوانیم روشی بکار ببریم که متغیرهای مشابه را در یک گروه قرار دهد، آنگاه دسته‌بندی افراد را می‌توان با گروه‌های متفاوتی از متغیرها (بجای استفاده از تک تک آنها) انجام داد. تجزیه به مؤلفه‌های اصلی تقریباً این کار را انجام می‌دهد و شاخص‌هایی از متغیرها (به نام مؤلفه‌های اصلی) ایجاد می‌کند که در هر شاخص، متغیرهای مشابه از وزن بالاتری برخوردار می‌باشند. سپس می‌توان از این شاخص‌ها برای گروه‌بندی افراد و بررسی روابط بین آنها، استفاده نمود. به طور مشابه، حجم زیادی از داده‌ها توسط نشانگرهای مولکولی ایجاد می‌شود که ممکن است بخشی از آنها جنبه مشابهی از افراد را نشان دهند. لذا تکنیک تجزیه به مؤلفه‌های اصلی در کاهش ابعاد داده‌های نشانگرهای مولکولی بسیار مفید خواهد بود (Jombart, 2008؛ Everitt and Dunn, 2001؛ Jolliffe, 1972؛ Jolliffe, 1973؛ Jolliffe, 1986؛ Jolliffe, 1989).

از بین روش‌های مختلفی که برای تجزیه داده‌های نشانگرهای مولکولی به کار می‌رود، روش‌های چندمتغیره مانند تجزیه به مؤلفه‌های اصلی از اهمیت ویژه‌ای برخوردار است زیرا این روش‌ها می‌توانند بدون نیاز به پیش‌فرض‌های الزامی برای مدل‌های تکاملی، داده‌های تنوع ژنتیکی را خلاصه‌سازی نماید. به عنوان مثال این روش‌ها به فرضیاتی همچون برقراری تعادل هاردی-واینبرگ یا غیاب پدیده عدم تعادل لینکاژی، متکی نیست (Jombart, 2008). این ویژگی در زمانی که اطلاعات ناچیزی از سیستمِ تحت مطالعه موجود است،

ارزشمند می‌باشد که خصوصاً در مطالعات ژنتیکی مربوط به جغرافیای طبیعی[18] امری معمول است (Manel *et al.*, 2003).

تجزیه به مؤلفه‌های اصلی را در نرم‌افزار R به روش‌ها و با استفاده از پکیج‌های مختلف می‌توان انجام داد. تابع dudi.pca در پکیج ade4 برای انجام این تجزیه مفید می‌باشد. نکته‌ای که حتماً باید مورد توجه قرار داد اینست که در صورت وجود مقادیر گمشده در داده‌ها، دستور dudi.pca عمل نمی‌کند و پیام خطا ظاهر می‌شود. لذا باید قبل از اجرای این تابع نسبت به رفع مقادیر گمشده اقدام کرد، که نحوه انجام این کار با استفاده از تابع scaleGen، در بخش مربوط به مقادیر گمشده توضیح داده شد.

در اینجا به عنوان مثال، نحوه کار با تابع dudi.pca برای انجام تجزیه به مؤلفه‌های اصلی بر روی مجموعه داده‌های microbov را شرح می‌دهیم:

```
> library(ade4)
> pca.cows<- dudi.pca(microbov)
    Error in dudi.pca(microbov) : na entries in table
```

پیام خطا نشان‌دهنده وجود مقادیر گمشده در داده‌ها است که با استفاده از تابع scaleGen به رفع این مشکل می‌پردازیم:

```
> x.cows<- scaleGen(microbov, NA.method="mean")
> pca.cows<- dudi.pca(x.cows, center=FALSE, scale=FALSE, scannf=FALSE, nf=3)
```

در تابع فوق آرگومان **center**، گزاره‌ای منطقی است که مشخص می‌کند آیا اعداد مربوط به شمارش آلل‌ها، کانونی (centered) شود یا خیر و آرگومان **scale**، گزاره‌ای منطقی است که مشخص می‌کند آیا اعداد مربوط به شمارش آلل‌ها، مقیاس (scaled) شود یا خیر. در هر دو آرگومان گزاره پیش‌فرض، TRUE قرار داده شده است. با توجه به اجرای تابع scaleGen در دستور قبلی، آرگومان scale را FALSE، قرار می‌دهیم. **nf**، عددی است که نشان‌دهندهٔ تعداد مؤلفه‌های اصلی است که بایستی نگه داشته شود. چنانچه مقدار آن را مشخص نکرده باشیم، یک نمودار screeplot از مقادیر ویژه (eigenvalues) ظاهر می‌شود و درخواست می‌کند که تعداد محورهایی که لازم است نگه داشته شود، مشخص شود.

```
> pca.cows
    Duality diagramm
    class: pca dudi
```

[18] Landscape genetics

```
$call: dudi.pca(df = x.cows, center = FALSE, scale = FALSE, scannf = FALSE,
   nf = 3)

$nf: 3 axis-components saved
$rank: 343
eigen values: 17.04 9.829 6.105 4.212 3.887 ...
  vector length mode    content
1 $cw    373    numeric column weights
2 $lw    704    numeric row weights
3 $eig   343    numeric eigen values

  data.frame nrow ncol content
1 $tab       704  373  modified array
2 $li        704  3    row coordinates
3 $l1        704  3    row normed scores
4 $co        373  3    column coordinates
5 $c1        373  3    column normed scores
other elements: cent norm
```

خروجی تابع dudi.pca ، "لیستی" متشکل از اجزای مختلف است اما سه جزء مهم از این لیست به قرار زیر می‌باشند:

$eig: وکتور عددی از مقادیر ویژه که نشان‌دهنده واریانس توجیه شده توسط هر مؤلفه اصلی است.

$li: قالب داده‌ای از مقادیر عددی مؤلفه‌های اصلی است که برای رسم نمودار پراکنش استفاده می‌شود. در این قالب داده، افراد در ردیف و مؤلفه‌های اصلی، در ستون قرار داده شده‌اند.

$c1: قالب داده‌ای از ضرایب مؤلفه‌های اصلی است و که مجذور آن نشان‌دهنده سهم هر آلل در هر مؤلفه اصلی می‌باشد.

```
> head(pca.cows$eig)
[1] 17.037969  9.828656  6.104656  4.212168  3.887299  3.664641
> head(pca.cows$li)
                 Axis1      Axis2      Axis3
AFBIBOR9503   8.868802   -6.12326   -0.55611
```

AFBIBOR9504	5.664997	-4.18074	0.171047
AFBIBOR9505	7.31954	1.757776	-1.36815
AFBIBOR9506	7.683053	-3.92465	-0.62767
AFBIBOR9507	8.253236	-1.73108	-1.09397
AFBIBOR9508	6.171968	1.915629	-0.49827

```
> head(pca.cows$c1)
```

	CS1	CS2	CS3
INRA63.167	0.008875	0.033367	0.015685
INRA63.171	0.006472	-0.01798	0.04324
INRA63.173	-0.01355	-0.00181	0.024708
INRA63.175	-0.14636	-0.03891	-0.03689
INRA63.177	0.018464	0.089306	0.079229
INRA63.179	-0.03421	-0.02216	-0.03108

می‌دانیم که مجموع مقادیر ویژه مؤلفه‌های اصلی باید برابر با مجموع تعداد صفات مورد بررسی باشد. در تجزیه به مؤلفه‌های اصلی بر روی اشیاء genind با استفاده از تابع dudi.pca، هر آلل در یک لوکوس، به عنوان یک متغیر در نظر گرفته می‌شود. بنابراین مجموع مقادیر ویژه مؤلفه‌های اصلی باید تقریباً برابر با مجموع تعداد متغیرهای اولیه باشد. می‌توانیم این موضوع را بررسی نماییم:

```
> head(pca.cows$eig)
    [1] 17.037969  9.828656  6.104656  4.212168  3.887299  3.664641
> sum(pca.cows$eig)
    [1] 372.4702
> sum(nAll(microbov))
    [1] 373
```

دلیل تفاوت جزئی بین دو مقدار، ناشی از گرد کردن اعداد اعشاری محاسبه شده می‌باشد.

همچنین می‌دانیم که مقدار ویژه هر مؤلفه اصلی برابر با واریانس آن مؤلفه اصلی می‌باشد. این موضوع را می‌توانیم امتحان کنیم. به عنوان مثال مقادیر مؤلفه اصلی اول با استفاده از دستور $li[,1]، قابل استخراج است. می‌توانیم با استفاده از تابع var، واریانس این مقادیر را محاسبه کنیم و با مقدار ویژه مؤلفه اصلی اول که در جزء eig[1] موجود است، مقایسه کنیم:

```
> head(pca.cows$li[,1])
    [1] 8.868802 5.664997 7.319540 7.683053 8.253236 6.171968
> var(pca.cows$li[,1])
```

```
[1] 17.0622
> pca.cows$eig[1]
[1] 17.03797
```

بنابراین می‌توانیم با استفاده از مقادیر ویژه‌ی هر مؤلفه اصلی، سهم آن را از واریانس کل به دست آورد:

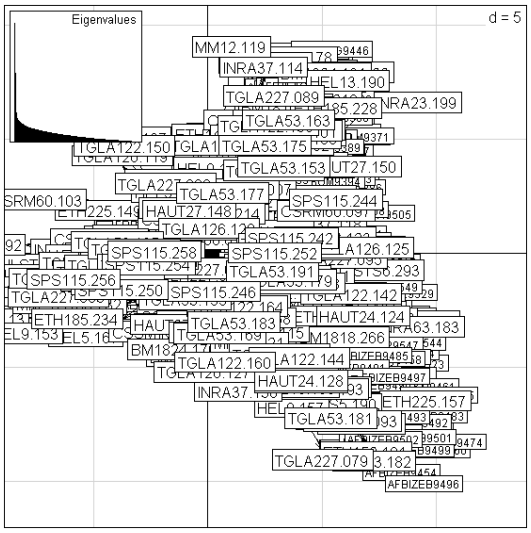
```
> eig.perc<- 100*pca.cows$eig/ sum(nAll(microbov))
> eig.perc[1:10]
[1] 4.5678200 2.6350284 1.6366369 1.1292677 1.0421714 0.9824774 0.9229985
[8] 0.9084639 0.8767267 0.8220948
```

براساس این نتایج، مؤلفه اصلی اول حدود ۴/۶ درصد از واریانس کل داده‌ها را شامل می‌شود.

ترسیم نمودار پراکنش براساس مؤلفه‌های اصلی

با استفاده از تابع scatter از پکیج ade4 برروی شیء خروجیِ تابع dudi.pca، می‌توان نمودار پراکنش آلل‌ها (شکل ۷-۱) را براساس مؤلفه‌های اصلی ترسیم نمود (ادامه از بخش قبل):

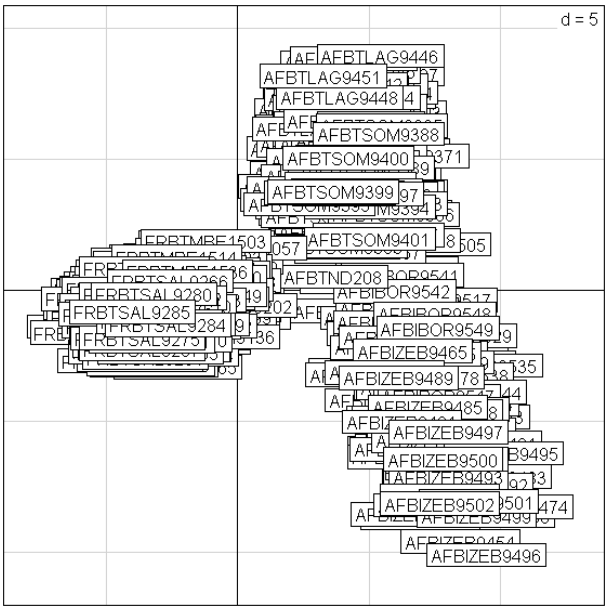
```
> scatter(pca.cows)
```



شکل ۷-۱ - تفکیک آلل‌ها در مجموعه داده microbov براساس تجزیه به مؤلفه‌های اصلی با استفاده از تابع scatter

همچنین با استفاده از تابع s.label از پکیج ade4، بر روی جزء li، از شیء خروجیِ تابع dudi.pca می‌توان نمودار پراکنش افراد را براساس مؤلفه‌های اصلی ترسیم نمود (شکل ۷-۲):

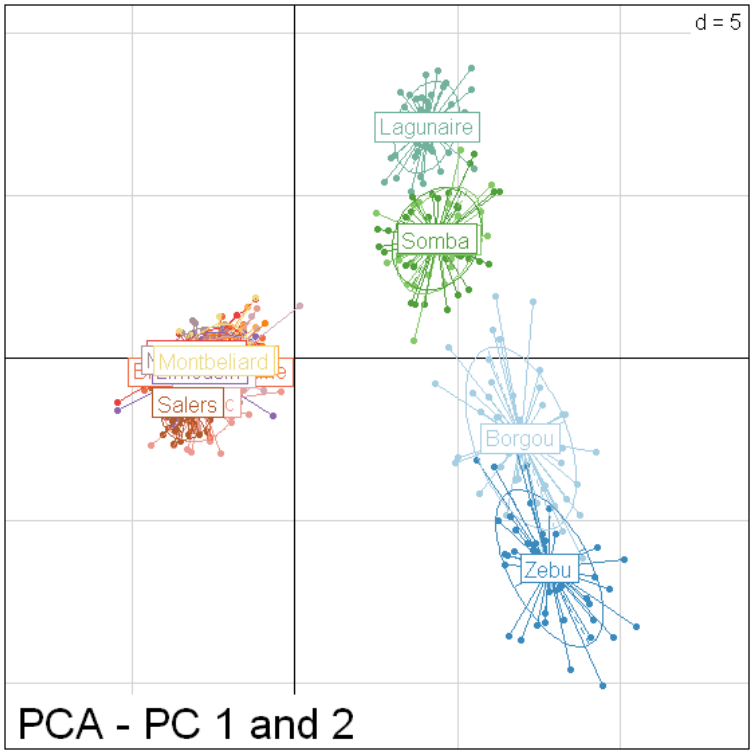
> s.label(pca.cows$li)



شکل ۷-۲ - تفکیک افراد مجموعه داده microbov براساس تجزیه به مؤلفه‌های اصلی با استفاده از تابع s.label

به منظور ایجاد وضوح و تفکیک بیشتر، بهتر است افراد را با در نظر گرفتن جمعیت انتسابی از یکدیگر متمایز نماییم. این کار با استفاده از تابع s.class از پکیج ade4، برروی جزء li، از شیء خروجیِ تابع dudi.pca قابل انجام است (شکل ۷-۳):

> s.class(pca.cows$li, xax = 1, yax = 2, fac=pop(microbov), col=funky(15), sub = "PCA - PC 1 and 2", csub = 2)

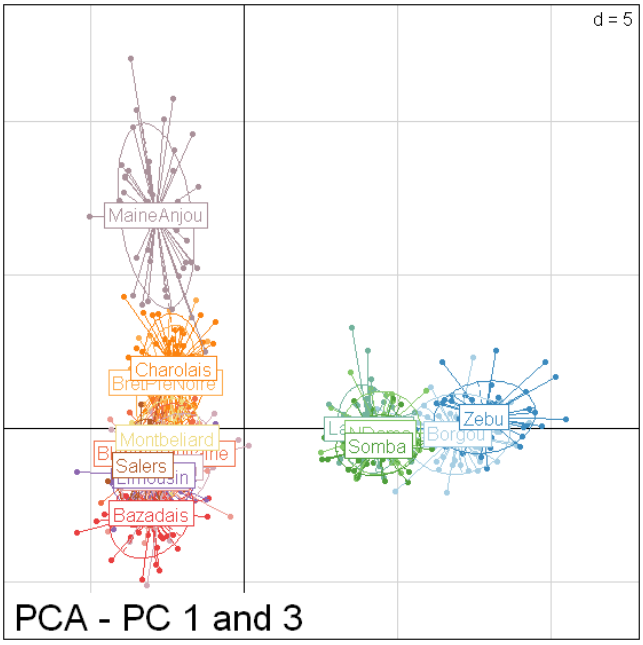


شکل ۷-۳– تفکیک افراد مجموعه داده microbov براساس دو مؤلفه اصلی اول با استفاده از تابع s.class

در این نمودار مشاهده می‌شود که چهار جمعیت Lagunaire، Somba، Borgou و Zebu براساس مؤلفه‌های اصلی اول و دوم، از سایر جمعیت‌ها تمایز یافته‌اند.

در دستور فوق، آرگومان csub، معیاری از اندازه برچسب راهنما است. همچنین با آرگومان‌های xax = 1 و yax = 2، مؤلفه‌های اصلی اول و دوم، به ترتیب به عنوان محور افقی و عمودی در ترسیم نمودار، مورد استفاده قرار گرفته‌اند. با استفاده از این آرگومان‌ها می‌توان مؤلفه‌های اصلی دلخواه را بصورت دو به دو استفاده نمود. به عنوان مثال برای ترسیم نمودار براساس مؤلفه اصلی اول و سوم خواهیم داشت (شکل ۷-۴):

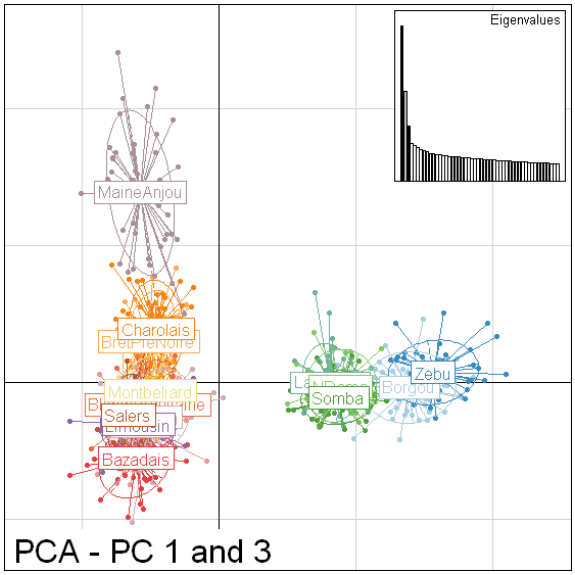
```
> s.class(pca.cows$li, xax = 1, yax = 3, fac=pop(microbov), col=funky(15), sub = "PCA - PC 1 and 3", csub = 2)
```



شکل ۷-۴ – تفکیک افراد مجموعه داده microbov براساس مؤلفه‌های اصلی اول و سوم با استفاده از تابع s.class

می‌توان از تابع add.scatter.eig، برای افزودن نمودار ستونی مقادیر ویژه به نمودار مؤلفه‌های اصلی، استفاده نمود (شکل ۷-۵):

```
> add.scatter.eig(pca.cows$eig[1:50],3,1,3, posi = "topright", ratio=0.3)
```



شکل ۷-۵ – تفکیک افراد مجموعه داده microbov براساس مؤلفه‌های اصلی اول و سوم با افزودن نمودار ستونی مقادیر ویژه

در دستور فوق، آرگومان posi محل قرار گرفتن نمودار ستونی مقادیر ویژه و آرگومان ratio، اندازه آن را نسبت به اندازه‌ی نمودار کلی تعیین می‌کند. می‌توان از گزینه‌های bottomleft، bottomright، topleft و topright برای آرگومان posi استفاده نمود.

مجذور ضرایب مؤلفه‌های اصلی، سهم هر آلل در مؤلفه مذکور را نشان می‌دهد. جزء c1 از شیء خروجیِ تابع dudi.pca شامل ضرایب مؤلفه‌های اصلی می‌باشد، لذا با استفاده از این اطلاعات به عنوان ورودی تابع loadingplot، می‌توان آلل‌های مؤثر در مؤلفه‌های اصلی را شناسایی نمود (شکل ۷-۶):

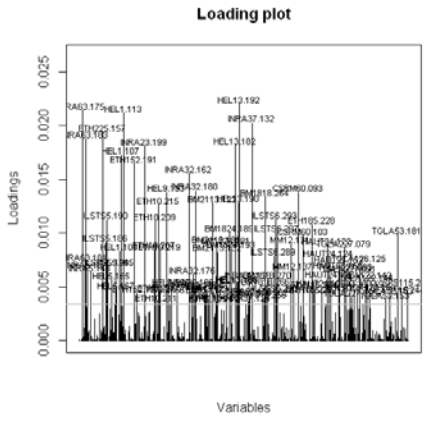
```
> head(pca.cows$c1)
```

	CS1	CS2	CS3
INRA63.167	0.008874894	0.033366808	0.01568508
INRA63.171	0.006471669	-0.017980601	0.04324038
INRA63.173	-0.013551673	-0.001814538	0.02470813
INRA63.175	-0.146362622	-0.038914387	-0.0368872
INRA63.177	0.018463914	0.089305776	0.07922882
INRA63.179	-0.034214781	-0.022161415	-0.03108483

```
> head(pca.cows$c1^2)
```

	CS1	CS2	CS3
INRA63.167	7.88E-05	1.11E-03	0.00024602
INRA63.171	4.19E-05	3.23E-04	0.00186973
INRA63.173	1.84E-04	3.29E-06	0.00061049
INRA63.175	2.14E-02	1.51E-03	0.00136067
INRA63.177	3.41E-04	7.98E-03	0.00627721
INRA63.179	1.17E-03	4.91E-04	0.00096627

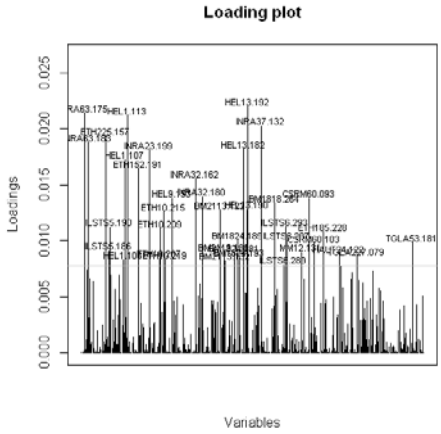
```
> loadingplot(pca.cows$c1^2)
```



شکل ۷-۶ – نمایش میزان تأثیر آلل‌ها روی تغییرات مؤلفه‌های اصلی در مجموعه داده microbov

می‌توان با تعریف یک آستانه توسط آرگومان threshold، آلل‌هایی که دارای نقش بیشتری هستند را متمایز نمود. مثلاً با دستور زیر، ۱۰ درصد از مؤثرترین آلل‌ها، قابل تمایز می‌باشند (شکل ۷-۷):

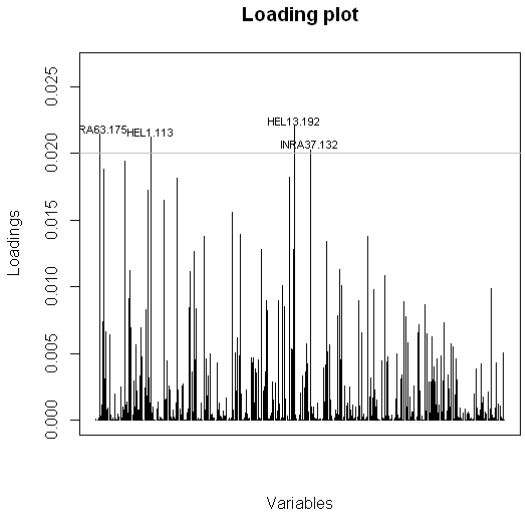
```
> p2<-pca.cows$c1^2
> lo.p2<-loadingplot(p2, threshold=quantile(p2,0.9))
```



شکل ۷-۷ – نمایش ده درصد از مؤثرترین آلل‌ها روی تغییرات مؤلفه‌های اصلی در مجموعه داده microbov

همچنین می‌توان آستانه را با مقدار عددی مشخص نمود (شکل ۷-۸):

```
> lo.p3<-loadingplot(p2, threshold=0.02)
```



شکل ۷-۸ – نمایش مؤثرترین آلل‌های مجموعه داده microbov در تغییرات ضرایب مؤلفه‌های اصلی با تعیین حد آستانه ۰/۰۲

اطلاعات مربوط به آلل‌های مؤثر براساس آستانه مشخص شده در خروجی تابع loadingplot موجود است:

```
> lo.p3
$threshold
[1] 0.02

$var.names
[1] "INRA63.175" "HEL1.113"   "HEL13.192"  "INRA37.132"

$var.idx
INRA63.175   HEL1.113  HEL13.192 INRA37.132
         4         51        182        196

$var.values
INRA63.175   HEL1.113  HEL13.192 INRA37.132
0.02142202 0.02118890 0.02204203 0.02027583
```

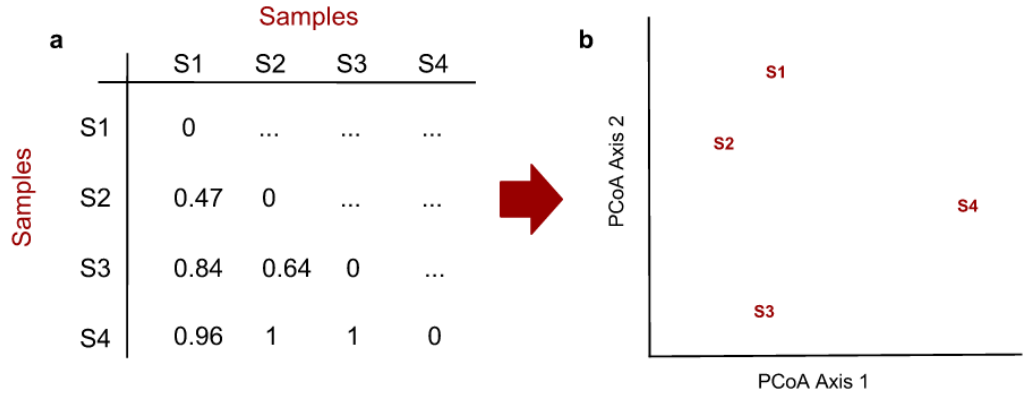
براساس این نتایج، آلل‌های INRA63.175، HEL1.113، HEL13.192 و INRA37.132 دارای بیشترین نقش در تغییرات مؤلفه‌های اصلی می‌باشند.

تجزیه به مختصات اصلی (PcoA)

تجزیه به مختصات اصلی[19] که به عنوان مقیاس‌بندی چندبعدی متریک[20] نیز شناخته می‌شود، روشی برای خلاصه کردن داده‌ها می‌باشد و با استفاده از آن می‌توان فواصل یا شباهت افراد را در فضای اقلیدسی نمایش داد (شکل ۷-۹). این تجزیه بجای داده‌های خام، ماتریس فاصله (یا شباهت) بین افراد را به عنوان ورودی می‌پذیرد و لذا درصورتی موفقیت‌آمیز خواهد بود که بتواند بخش عمده‌ی تغییرات موجود در ماتریس فاصله (یا شباهت) را به تعداد محدودی محور مختصات کاهش دهد. لذا تجزیه به مختصات اصلی از این جهت (کاهش ابعاد داده‌ها) مشابه با تجزیه به مؤلفه‌های اصلی و تجزیه تطبیقی می‌باشد، با این تفاوت که تجزیه به مؤلفه‌های اصلی و تجزیه تطبیقی بترتیب مبتنی بر فواصل اقلیدسی بین افراد و χ^2 می‌باشند، ولی در تجزیه

[19] Principal Coordinates Analysis
[20] Metric Multidimensional Scaling

به مختصات اصلی، هرگونه کمیّت شباهت (یا عدم شباهت) قابل استفاده است که سبب می‌شود این روش تجزیه، از انعطاف‌پذیری و درنتیجه، دامنه کاربرد وسیع‌تری، برخوردار شود (Legendre and Legendre, 1998؛ Gower, 1966؛ Buttigieg and Ramette, 2014).



شکل ۷-۹- تجزیه به مختصات اصلی براساس ماتریس فاصله. افرادی که در نمودار نزدیکتر به یکدیگر واقع شده‌اند مقدار عددی کوچکتری در ماتریس فاصله دارند (برگرفته از Buttigieg and Ramette, 2014).

تجزیه به مختصات اصلی در R با استفاده از تابع dudi.pcao از پکیج ade4، قابل انجام است. این تابع به عنوان ورودی ماتریس فاصله را می‌پذیرد، لذا می‌توان تجزیه به مختصات اصلی را براساس فواصل بین افراد یا بین جمعیت‌ها، انجام داد. روش انجام این تجزیه در ذیل به تفکیک براساس افراد و جمعیت‌ها، شرح داده می‌شود.

تجزیه به مختصات اصلی مبتنی بر افراد

برای انجام تجزیه به مختصات اصلی مبتنی بر افراد، همانطور که اشاره شد ابتدا لازم است فواصل ژنتیکی بین افراد محاسبه شود. برای این کار از توابعی که در بخش مربوط به محاسبه فواصل ژنتیکی شرح داده شد می‌توان استفاده کرد. اما علاوه بر توابع مذکور سایر معیارهای فاصله بین افراد نیز قابل استفاده است. به عنوان مثال در اینجا از تابع propShared که قبلاً ذکر شد، استفاده می‌کنیم. یادآوری می‌شود که تابع propShared نسبت آلل‌های مشترک بین افراد را محاسبه می‌کند. بنابراین می‌تواند به عنوان معیاری از شباهت بین افراد مد نظر قرار گیرد (یعنی افرادی که آلل‌های مشترک بیشتری دارند شباهت بیشتری خواهند داشت و بالعکس). با ضرب کردن ماتریس خروجی این تابع در ۱- و توان قرار دادن اعداد حاصل

برپایه عدد نپر (e= 2.718) با استفاده از تابع exp، می‌توان ماتریس شباهت را به ماتریس فاصله تبدیل کرد. عملیات فوق به عنوان مثال برای مجموعه داده microbov، بصورت زیر خواهد بود:

```
> library(adegenet)
> data(microbov)
> mic.sim <- propShared(microbov)
> mic.sim[1:10,1:5]
```

	AFBIBOR9503	AFBIBOR9504	AFBIBOR9505	AFBIBOR9506	AFBIBOR9507
AFBIBOR9503	1	0.433333	0.266667	0.466667	0.333333
AFBIBOR9504	0.433333	1	0.383333	0.366667	0.383333
AFBIBOR9505	0.266667	0.383333	1	0.35	0.333333
AFBIBOR9506	0.466667	0.366667	0.35	1	0.316667
AFBIBOR9507	0.333333	0.383333	0.333333	0.316667	1
AFBIBOR9508	0.383333	0.366667	0.433333	0.45	0.5
AFBIBOR9509	0.275862	0.362069	0.448276	0.37931	0.431035
AFBIBOR9510	0.416667	0.366667	0.35	0.366667	0.466667
AFBIBOR9511	0.4	0.4	0.5	0.416667	0.383333
AFBIBOR9512	0.35	0.383333	0.316667	0.4	0.333333

```
> class(mic.sim)
[1] "matrix"
> dim(mic.sim)
[1] 704 704
> mic.dis <- exp(-mic.sim)
> for(i in 1: nrow(mic.dis)){mic.dis[i,i]=0}
> mic.dis[1:10,1:5]
```

	AFBIBOR9503	AFBIBOR9504	AFBIBOR9505	AFBIBOR9506	AFBIBOR9507
AFBIBOR9503	0	0.648344	0.765928	0.627089	0.716531
AFBIBOR9504	0.648344	0	0.681586	0.693041	0.681586
AFBIBOR9505	0.765928	0.681586	0	0.704688	0.716531
AFBIBOR9506	0.627089	0.693041	0.704688	0	0.728574
AFBIBOR9507	0.716531	0.681586	0.716531	0.728574	0
AFBIBOR9508	0.681586	0.693041	0.648344	0.637628	0.606531
AFBIBOR9509	0.758918	0.696234	0.638729	0.684333	0.649837
AFBIBOR9510	0.659241	0.693041	0.704688	0.693041	0.627089
AFBIBOR9511	0.67032	0.67032	0.606531	0.659241	0.681586
AFBIBOR9512	0.704688	0.681586	0.728574	0.67032	0.716531

```
> class(mic.dis)
        [1] "matrix"
> mic.dis<-as.dist(mic.dis)
> is.euclid(mic.dis)
        [1] FALSE
```

همانطور که مشاهده می‌شود ماتریس فاصله‌ی ایجاد شده، از نوع اقلیدسی نیست لذا از تابع cailliez برای تبدیل فواصل غیراقلیدسی به فواصل اقلیدسی استفاده می‌کنیم:

```
> D <- cailliez(mic.dis)
> is.euclid(D)
        [1] TRUE
```

اکنون می‌توان تجزیه به مختصات اصلی را انجام داد:

```
> mic.pcoa <- dudi.pco(D, scannf=FALSE, nf=3)
> mic.pcoa
        Duality diagramm
        class: pco dudi
        $call: dudi.pco(d = D, scannf = FALSE, nf = 3)

        $nf: 3 axis-components saved
        $rank: 702
        eigen values: 0.0673 0.02793 0.02259 0.01533 0.01355 ...
          vector length mode    content
        1 $cw    702    numeric column weights
        2 $lw    704    numeric row weights
        3 $eig   702    numeric eigen values

          data.frame nrow ncol content
        1 $tab       704  702  modified array
        2 $li        704  3    row coordinates
```

3 $l1 704 3 row normed scores

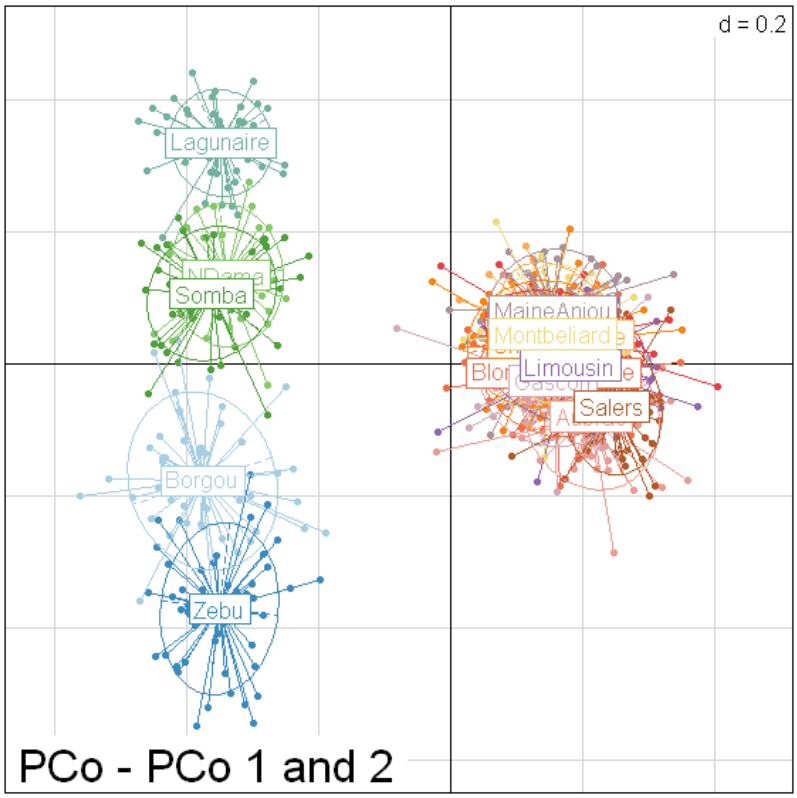
4 $co 702 3 column coordinates

5 $c1 702 3 column normed scores

other elements: NULL

با توجه به اینکه اجزاء شیء خروجیِ تابع dudi.pco تا حد زیادی مشابه با اشیاء خروجیِ تابع dudi.pca می‌باشند، از توضیح بیشتر در مورد آنها خودداری می‌شود. بطور مشابه با تابع dudi.pca، می‌توان از جزء li (مقادیر عددی مختصات اصلی)، برای ترسیم نمودار پراکنش افراد استفاده نمود (شکل ۷-۱۰):

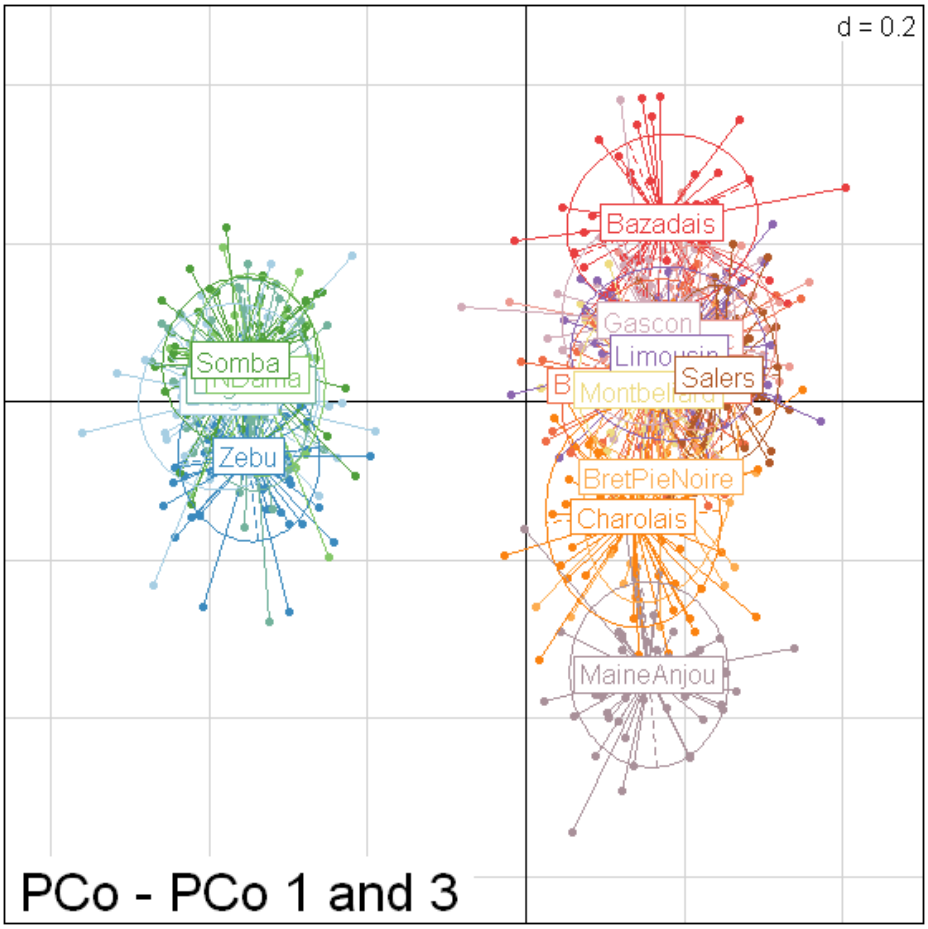
```
> s.class(mic.pcoa$li, xax = 1, yax = 2, fac=pop(microbov), col=funky(15), sub = "PCo - PCo 1 and 2", csub = 2)
```



شکل ۷-۱۰ – تفکیک افراد براساس محورهای اول و دوم مربوط به تجزیه به مختصات اصلی در مجموعه داده microbov

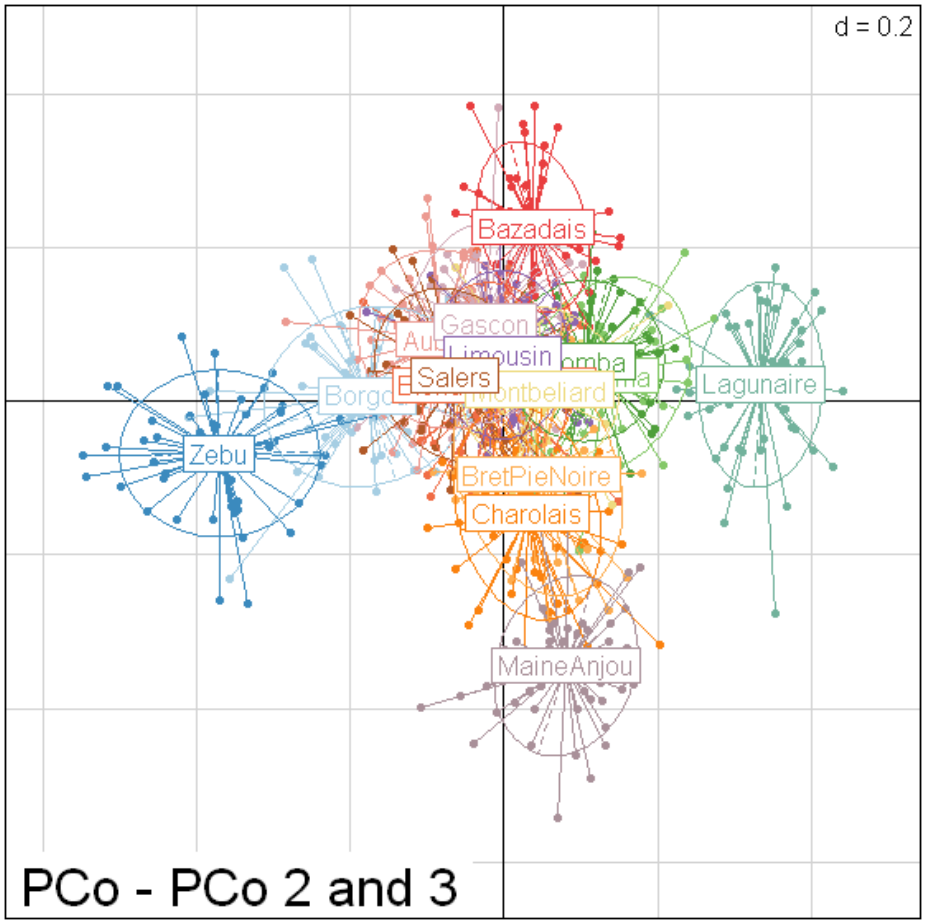
بطور مشابه نمودار پراکنش افراد براساس محورهای مختصات اول و سوم (شکل ۷-۱۱) و همچنین براساس محورهای مختصات دوم و سوم به صورت زیر خواهد بود (شکل ۷-۱۲):

> s.class(mic.pcoa$li, xax = 1, yax = 3, fac=pop(microbov), col=funky(15), sub = "PCo - PCo 1 and 3", csub = 2)



شکل ۷-۱۱ – تفکیک افراد براساس محورهای اول و سوم مربوط به تجزیه به مختصات اصلی در مجموعه داده microbov

> s.class(mic.pcoa$li, xax = 2, yax = 3, fac=pop(microbov), col=funky(15), sub = "PCo - PCo 2 and 3", csub = 2)



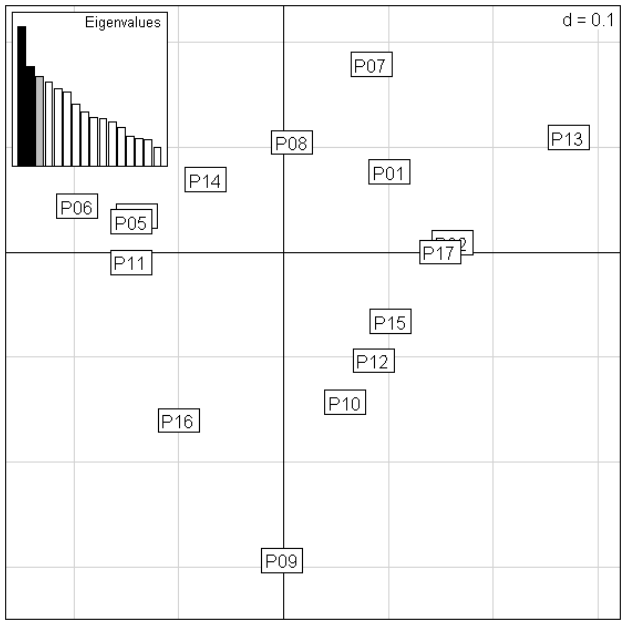
شکل ۷-۱۲– تفکیک افراد براساس محورهای دوم وسوم مربوط به تجزیه به مختصات اصلی در مجموعه داده microbov

تجزیه به مختصات اصلی مبتنی بر جمعیت‌ها

برای شرح نحوه‌ی تجزیه به مختصات اصلی براساس جمعیت‌ها به عنوان مثال، فواصل بین جمعیت‌ها در مجموعه داده nancycats را به روش Edwards توسط تابع dist.genpop، محاسبه نموده و تابع dudi.pco را برای انجام تجزیه مورد استفاده قرار می‌دهیم (شکل ۷-۱۳). برای جزئیات بیشتر در مورد تابع dist.genpop و انواع فواصل، به بخش مربوط به محاسبه فواصل ژنتیکی بین جمعیت‌ها مراجعه نمایید.

```
> library(adegenet)
> data(nancycats)
> p.cats<-genind2genpop(nancycats)
>  d2.p.cats<-dist.genpop(p.cats, met=2)
```

```
> class(d2.p.cats)
        [1] "dist"
> d2.p.cats.pcoa <- dudi.pco(d2.p.cats,scannf=FALSE,nf=3)
> scatter(d2.p.cats.pcoa)
```



شکل ۷-۱۳– تفکیک جمعیت‌های مجموعه داده nancycats توسط تجزیه به مختصات اصلی براساس معیار فاصله Edwards

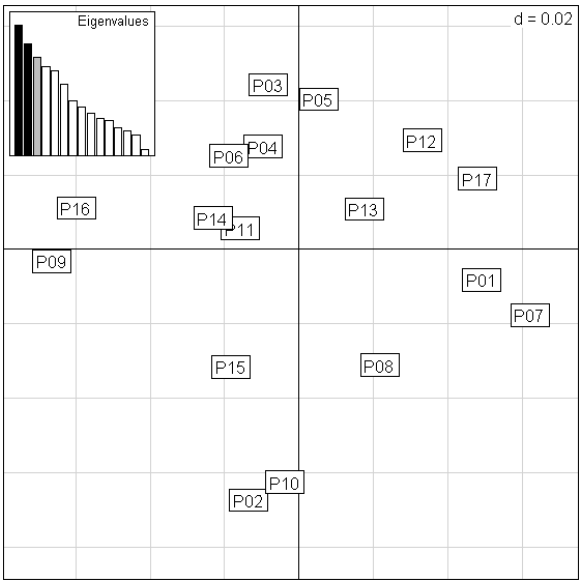
همانطور که در بخش مربوط به آماره‌های افتراق ژنتیکی اشاره شد، مقادیر دو به دوی F_{ST} بین جمعیت‌ها نیز معیاری از فاصله بین آنها به شمار می‌رود. لذا می‌توان تجزیه فوق را براساس این کمّیت نیز به صورت زیر انجام داد و جمعیت‌ها را براساس محورهای مختصات اصلی تفکیک نمود (شکل ۷-۱۴):

```
> library(hierfstat)
>  data(nancycats)
> cats.pdis<- pairwise.fst(nancycats,res.type="matrix")
> class(cats.pdis)
        [1] "matrix"
> dim(cats.pdis)
        [1] 17 17
```

```
> cats.pdis[1:10,1:5]
```

	P01	P02	P03	P04	P05
P01	0	0.080185	0.071408	0.049925	0.059069
P02	0.080185	0	0.082009	0.069855	0.082953
P03	0.071408	0.082009	0	0.025716	0.053047
P04	0.049925	0.069855	0.025716	0	0.038335
P05	0.059069	0.082953	0.053047	0.038335	0
P06	0.078202	0.064061	0.038258	0.034501	0.060683
P07	0.061229	0.070468	0.076064	0.055229	0.084264
P08	0.057779	0.055715	0.058387	0.034498	0.066616
P09	0.087841	0.07205	0.080486	0.048394	0.086531
P10	0.057257	0.053137	0.075677	0.062283	0.069681

```
> cats.pdis<-as.dist(cats.pdis)

> is.euclid(cats.pdis)

        [1] FALSE

> D <- cailliez(cats.pdis)

> is.euclid(D)

        [1] TRUE

> cats.pcoa <- dudi.pco(D, scannf=FALSE,nf=3)

>  scatter(cats.pcoa)
```



شکل ۷-۱۴ – تفکیک جمعیت‌های مجموعه داده nancycats توسط تجزیه به مختصات اصلی براساس آماره F_{ST} به عنوان معیار فاصله

تجزیه تطبیقی

تجزیه تطبیقی[۲۱] تکنیکی آماری است که به منظور نمایش گرافیکی جداول متقاطع، که تحت عنوان جداول توافق[۲۲] نیز شناخته می‌شوند، بکار می‌رود. این جداول اغلب در زمانی تشکیل می‌شوند که بتوان داده‌ها را از دو جهتِ متفاوت، طبقه‌بندی نمود. در زندگی روزمره مثال‌های متعددی از اینگونه طبقه‌بندی می‌توان یافت. به عنوان مثال می‌توان رفتار یا گرایشی خاص و یا دچار شدن به نوعی عارضه را برحسب جنسیت افراد، بررسی نمود. در اینحالت افراد از سویی برحسب ابتلاء یا عدم ابتلاء به عارضه و یا نشان دادن رفتاری خاص یا عدم آن و از سوی دیگر برحسب جنسیت طبقه‌بندی می‌شوند. دراینصورت با انجام آزمون آماری (مانند χ^2) می‌توان پی برد که آیا جنسیت افراد با گرایش رفتاری خاص یا دچار شدن به عارضه‌ی مورد بررسی، در ارتباط می‌باشد و یا خیر. به همین ترتیب، داده‌های نشانگرهای ژنتیکی نیز از طبقه‌بندی دوسویه برخوردار می‌باشند زیرا این داده‌ها از یک جهت برحسب افراد یا جمعیت‌های متفاوت و از جهت دیگر، برحسب نشانگرهای ژنتیکیِ استفاده شده، قابل طبقه‌بندی می‌باشند. تجزیه تطبیقی ارتباط بین این متغیرهای گروهی را در قالب نمودار یا نقشه، قابل نمایش می‌سازد. لذا با استفاده از این تجزیه، می‌توان الگوها یا ساختارهای موجود در جمعیت را بصورت چشمی بررسی نمود. جنبه مهم تجزیه تطبیقی که آن را از سایر روش‌های مرسوم متمایز می‌نماید اینست که روشی تأییدی نیست، یعنی تلاش ندارد که فرضی را اثبات کند، بلکه تکنیکی اکتشافی می‌باشد. می‌توان گفت که تجزیه تطبیقی پنجره‌ای به روی داده‌ها می‌گشاید، دسترسی آسان‌تر محققان را به نتایج عددی تحقیق‌شان میسر ساخته و بحث پیرامون داده‌ها و احتمالاً فرضیه‌سازی برای انجام آزمون‌های بعدی را تسهیل می‌بخشد. این روش از لحاظ ریاضی، χ^2 محاسبه شده در یک جدول توافق را به اجزاء تشکیل دهنده‌اش تجزیه می‌نماید. در تجزیه تطبیقی سعی براین است که با انتساب مقیاسی به ردیف‌ها و مقیاسی جداگانه به ستون‌ها، همبستگی بین جفت متغیرهای حاصل به حداکثر رسانیده شود. باید بخاطر داشت که به تجزیه تطبیقی بایستی به عنوان روشی تکمیلی برای سایر تجزیه‌های تحلیلی که برای متغیرهای گروهی بکار می‌روند و نه به عنوان جایگزینی برای آن‌ها، نگریسته شود (Hill, 1974؛ Hill and, 1980؛ Leeuw, 1983؛ Greenacre, 1992؛ Benzecri, 1992؛ Everitt and Dunn, 2001؛ Greenacre, 2007؛ Lebart, 2013).

تجزیه تطبیقی با استفاده از تابع dudi.coa از پکیج ade4، قابل انجام است:

dudi.coa(df, scannf = TRUE, nf = 2)

[21] Correspondence Analysis
[22] Contingency tables

آرگومان **df**، قالب داده حاوی مقادیر مثبت یا صفر، می‌باشد. آرگومان **scannf**، گزاره‌ای منطقی است که درصورتِ انتخابِ گزینه FALSE برای آن، نمودار ستونی مربوط به مقادیر ویژه، نشان داده نخواهد شد. این آرگومان بطور پیش‌فرض، TRUE می‌باشد. چنانچه گزینه FALSE برای scannf انتخاب شده، در اینصورت با قرار دادن مقداری عددی برای آرگومان **nf**، می‌توان تعداد محورهای تجزیه تطبیقی که باید نگه داشته شود را تعیین نمود.

در اینجا به عنوان مثال، تجزیه تطبیقی را برای جمعیت‌های مجموعه داده microbov انجام می‌دهیم. بدین منظور ابتدا با استفاده از تابع genind2genpop شیء genind را به شیء genpop تبدیل می‌نماییم. سپس با توجه با اینکه تابع dudi.coa یک قالب داده (آرگومان df) را به عنوان ورودی می‌پذیرد، با استفاده از تابع tab داده‌های microbov را در تابع dudi.coa قرار می‌دهیم:

```
> library(adegenet)
> data(microbov)
> mic.pop <- genind2genpop(microbov)
> mic.ca <- dudi.coa(tab(mic.pop),scannf=FALSE,nf=3)
> mic.ca
        Duality diagramm
        class: coa dudi
        $call: dudi.coa(df = tab(mic.pop), scannf = FALSE, nf = 3)

        $nf: 3 axis-components saved
        $rank: 14
        eigen values: 0.2776 0.1558 0.08948 0.05055 0.04669 ...
          vector length mode    content
        1 $cw    373    numeric column weights
        2 $lw    15     numeric row weights
        3 $eig   14     numeric eigen values

          data.frame nrow ncol content
        1 $tab       15   373  modified array
        2 $li        15   3    row coordinates
        3 $l1        15   3    row normed scores
        4 $co        373  3    column coordinates
        5 $c1        373  3    column normed scores
        other elements: N
```

شیء خروجی تابع dudi.coa از اجزاء مختلف تشکیل شده است که در مقابل هر یک، توضیحی ارائه شده است. این اجزاء با علامت $ قابل فراخوانی می‌باشند. به عنوان مثال برای مشاهده مقادیر ویژه و مختصات جمعیت‌ها براساس محورهای تجزیه تطبیقی، می‌توان بترتیب اجزاء eig و li را فراخوانی نمود:

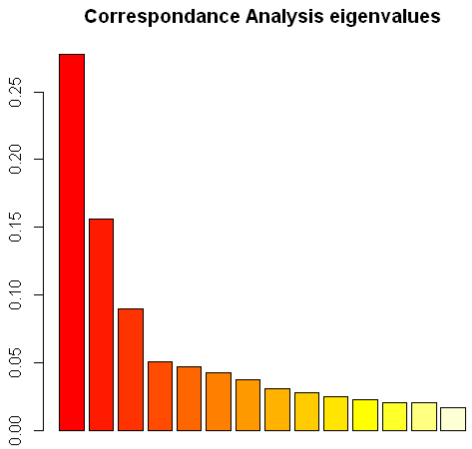
```
> mic.ca$eig
        [1] 0.27759423 0.15583551 0.08948399 0.05054847 0.04669340 0.04237420
        [7] 0.03719634 0.03053873 0.02791541 0.02503496 0.02304518 0.02081760
       [13] 0.02032391 0.01661315
```

> mic.ca$li

	Axis1	Axis2	Axis3
Borgou	-0.92752	0.272984	0.007154
Zebu	-1.10578	0.781396	0.046005
Lagunaire	-0.44382	-0.96832	0.014948
NDama	-0.5052	-0.51077	-0.00079
Somba	-0.48957	-0.52023	-0.04817
Aubrac	0.331877	0.178746	-0.23027
Bazadais	0.384724	0.024849	-0.31201
BlondeAquitaine	0.331592	0.070804	-0.13228
BretPieNoire	0.346455	0.026878	0.193266
Charolais	0.333089	0.050305	0.258672
Gascon	0.311721	0.07043	-0.16621
Limousin	0.368617	0.060629	-0.22514
MaineAnjou	0.416915	0.067086	0.963117
Montbeliard	0.342497	0.039914	-0.03367
Salers	0.382732	0.190838	-0.2065

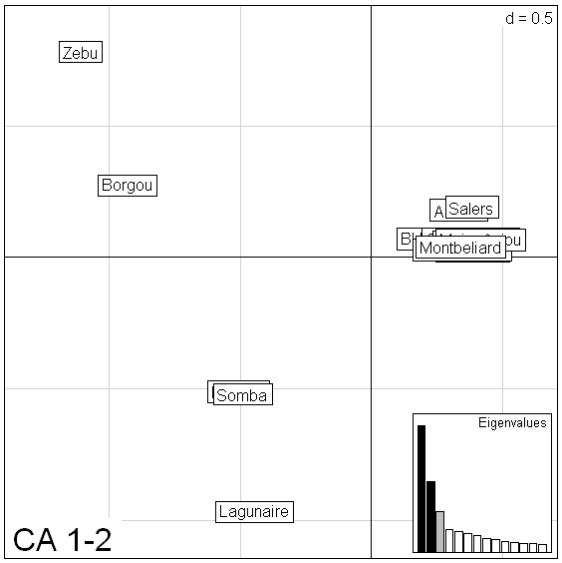
با توجه به اینکه اجزاء شیء خروجی تابع dudi.coa تا حد زیادی مشابه با اشیاء خروجی تابع dudi.pca می‌باشند، از توضیح بیشتر در مورد آن خودداری می‌شود. بطور مشابه با تابع dudi.pca، می‌توان از اجزاء eig و li (مربوط به مقادیر ویژه و مختصاتِ محورهای تجزیه تطبیقی)، بترتیب برای ترسیم نمودار ستونی مقادیر ویژه (شکل ۷-۱۵) و نمودار پراکنش اعضاء (شکل ۷-۱۶) استفاده نمود:

> barplot(mic.ca$eig,main="Correspondance Analysis eigenvalues", col=
heat.colors(length(ca1$eig)))



شکل ۷-۱۵ – نمودار ستونی مقادیر ویژه تجزیه تطبیقی در مجموعه داده microbov

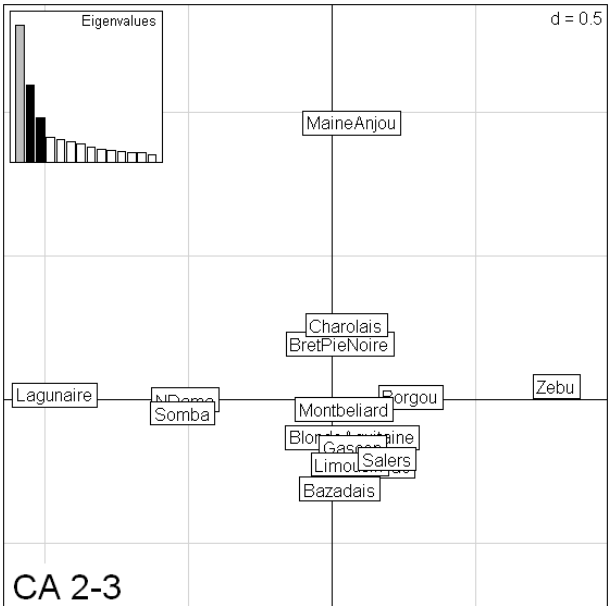
```
> s.label(mic.ca$li, sub="CA 1-2",csub=2)
> add.scatter.eig(mic.ca $eig,nf=3,xax=1,yax=2,posi="bottomright")
```



شکل ۷-۱۶ – تفکیک جمعیت‌های مجموعه داده microbov براساس محورهای اول و دوم تجزیه تطبیقی

همانطور که مشاهده می‌شود جمعیت‌های مجموعه داده microbov براساس محورهای اول و دوم تجزیه تطبیقی بخوبی از یکدیگر تفکیک شده‌اند. چنانچه بخواهیم جمعیت‌ها براساس محورهای دوم و سوم تجزیه تطبیقی تفکیک شوند (شکل ۷-۱۷)، دستورات زیر را اجرا می‌نماییم:

```
> s.label(mic.ca$li,xax=2,yax=3,lab=popNames(mic.pop),sub="CA 2-3",csub=2)
> add.scatter.eig(mic.ca$eig,nf=3,xax=2,yax=3,posi="topleft")
```



شکل ۷-۱۷ – تفکیک جمعیت‌های مجموعه داده microbov براساس محورهای دوم و سوم تجزیه تطبیقی

تجزیه کلاستر به روش K means

تجزیه کلاستر K means روشی برای دسته‌بندی مشاهدات است که برای نخستین بار توسط مک‌کوئین (MacQueen, 1967) استفاده شد و هدف از آن انتساب افراد به K گروه مختلف می‌باشد. نحوه انجام این تجزیه بدین صورت است که ابتدا تعداد مشخصی گروه (K) در نظر گرفته می‌شود و افراد در گروهی قرار می‌گیرند که نزدیکترین فاصله را با کانون آن گروه داشته باشند. سپس تعداد گروه‌ها افزایش یافته و مجدداً انتساب افراد به گروه‌ها تکرار می‌شود. این فرایند تا آنجا پیش می‌رود که به حداکثر تفاوت بین گروهی بیانجامد. برای انجام تجزیه کلاستر K means به عنوان مثال، از مجموعه داده dapcIllus موجود در پکیج adegenet استفاده می‌نماییم. داده‌های dapcIllus، خود لیستی مشتمل بر چهار مجموعه داده به نام‌های a، b، c و d است که برای اهداف مختلف شبیه‌سازی شده‌اند. برای انجام تجزیه کلاستر K means، مجموعه داده‌های a را مورد استفاده قرار می‌دهیم:

```
> library(adegenet)
> data(dapcIllus)
> class(dapcIllus)
        [1] "list"
> names(dapcIllus)
        [1] "a" "b" "c" "d"
> summary(dapcIllus)
          Length Class  Mode
        a 1      genind S4
        b 1      genind S4
        c 1      genind S4
        d 1      genind S4
> x <- dapcIllus$a
> x
        /// GENIND OBJECT /////////
         // 600 individuals; 30 loci; 140 alleles; size: 379.4 Kb
         // Basic content
           @tab:  600 x 140 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 2-8)
```

@loc.fac: locus factor for the 140 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: read.fstat(file = file, missing = missing, quiet = quiet)

// Optional content

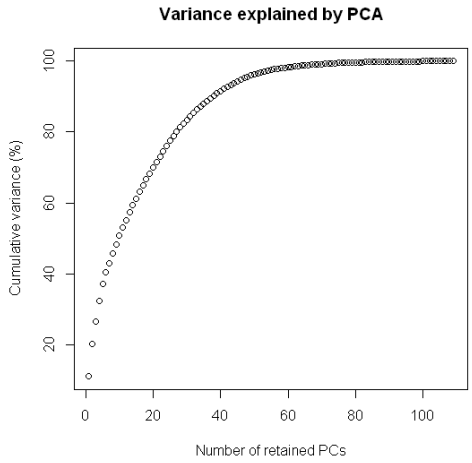
@pop: population of each individual (group size range: 100-100)

تجزیه کلاستر K means با استفاده از تابع find.clusters در پکیج adegenet قابل انجام می‌باشد. این تابع، تجزیه کلاستر را ابتدا با تبدیل داده‌ها براساس تجزیه به مؤلفه‌های اصلی و سپس با در نظر گرفتن تعداد مختلف کلاستر(K) بصورت افزایشی، انجام می‌دهد. بدین منظور به ازای هر تعداد کلاستر (K)، آماره‌ای به نام BIC[۲۳] محاسبه می‌شود. مقدار این آماره، معیاری از تفاوت بین کلاسترها می‌باشد بطوری که هرچه مقدار این آماره کمتر باشد، نشان‌دهنده‌یِ وجودِ تفاوتِ بیشتر، بین کلاسترها خواهد بود. بنابراین بهترین تعداد کلاستر (K) با کمترین مقدار BIC، مشخص می‌شود. از آنجا که انجام این الگوریتم برای انواع ممکنِ تعدادِ کلاستر، ممکن است به طول بیانجامد می‌توان با استفاده از آرگومان max.n.clust رویه‌ی افزایشِ تعدادِ کلاسترها را تا تعداد معینی از K، محدود نماییم:

```
> grp <- find.clusters(x, max.n.clust=40)
```

با اجرای دستور فوق، نمودار درصد واریانس تجمعی توجیه شده توسط مؤلفه‌های اصلی، ظاهر شده (شکل ۷-۱۸) و از کاربر درخواست می‌شود که تعدادِ مؤلفه‌های اصلی که تجزیه کلاستر بایستی برمبنای آن انجام شود را، مشخص نماید.

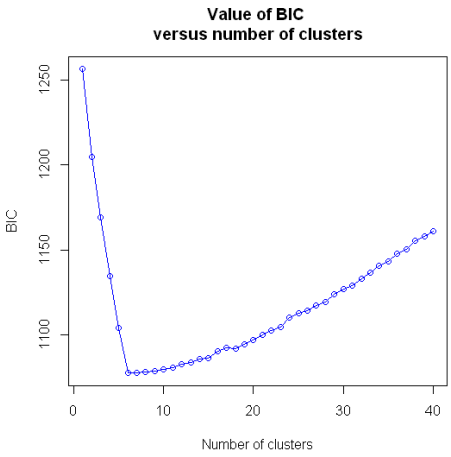
[23] Bayesian Information Criterion



شکل ۷-۱۸ - درصد واریانس تجمعی توجیه شده توسط مؤلفه‌های اصلی روی داده‌های a از مجموعه dapcIllus در مرحله اول از تجزیه کلاستر به روش K means

از آنجا که قصد داریم از تمام اطلاعات موجود استفاده نماییم، بهتر است مقدار عددی را وارد نماییم که همه مؤلفه‌هایِ اصلیِ ممکن را پوشش دهد. حداکثر تعدادِ مؤلفه‌های اصلی، به تعدادِ آلل‌ها می‌باشد و از آنجا که ۱۴۰ آلل در مجموعه داده‌های فوق موجود است پس بایستی عددی برابر یا بزرگتر از ۱۴۰ وارد کنیم (هرچند که در نمودار نیز مشخص است که از مؤلفه اصلی ۹۰اُم به بعد افزایش قابل توجهی در واریانس تجمعی مشاهده نمی‌شود). مثلاً در پاسخ به دستور فوق عدد ۱۵۰ را وارد نمایید:

Choose the number PCs to retain (>=1): 150



شکل ۷-۱۹ – روند تغییرات مقدار آماره BIC به ازای تعداد مختلف کلاستر (K) در تجزیه کلاستر به روش K means برای داده‌های a از مجموعه dapcIllus

در نمودار فوق تغییرات مقادیر BIC به ازای تعداد متفاوت کلاستر (K) نشان داده شده است. همانطور که مشاهده می‌شود، در حدود K=۶، مقدار BIC، حداقل می‌باشد. بنابراین در پاسخ به سؤال تعداد کلاستر (Choose the number of clusters)، عدد ۶ را وارد کنید:

```
Choose the number of clusters (>=2: 6
```

خروجی تابع ، لیستی متشکل از اجزاء مختلف است:

```
> class(grp)
[1] "list"
> names(grp)
[1] "Kstat" "stat"  "grp"   "size"
```

جزء Kstat، وکتوری است که مقادیر BIC، به ازای تعداد متفاوت کلاسترها را نشان می‌باشد و جزء stat مقدار BIC را در K انتخابی نشان می‌دهد:

```
> grp$Kstat
```

K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8
1256.185	1204.763	1169.134	1134.633	1104.379	1078.113	1077.659	1078.416
K=9	K=10	K=11	K=12	K=13	K=14	K=15	K=16
1079.096	1080.025	1080.814	1082.767	1084.016	1085.958	1086.771	1090.518
K=17	K=18	K=19	K=20	K=21	K=22	K=23	K=24
1092.38	1092.069	1094.889	1097.248	1100.215	1102.981	1104.84	1110.331
K=25	K=26	K=27	K=28	K=29	K=30	K=31	K=32
1113.046	1114.485	1117.314	1119.384	1124.162	1127.011	1128.916	1133.325
K=33	K=34	K=35	K=36	K=37	K=38	K=39	K=40
1136.76	1140.875	1143.517	1147.918	1150.342	1155.347	1158.055	1161.272

```
> grp$stat
      K=6
1078.113
```

جزء grp، کلاستر انتسابی افراد و جزء size، اندازه هر یک از کلاسترها (تعداد افراد عضو) را نشان می‌دهد:

```
> head(grp$grp, 10)
 1  2  3  4  5  6  7  8  9 10
 1  1  1  4  1  1  1  1  1  1
Levels: 1 2 3 4 5 6
```

```
> tail(grp$grp, 10)

 591  592  593  594  595  596  597  598  599  600
   4    4    4    4    4    4    4    4    4    4
Levels: 1 2 3 4 5 6

> grp$size

[1]  98 102  99 105  99  97
```

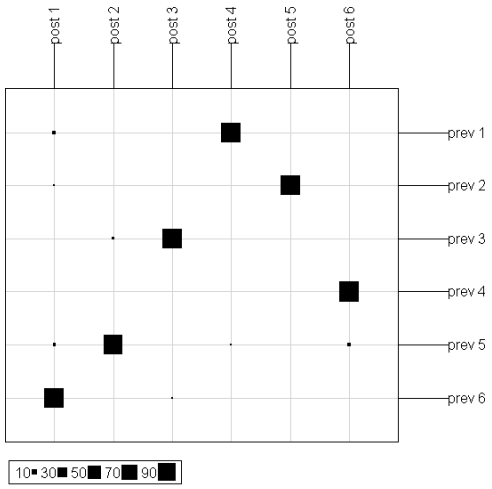
با استفاده از تابع table، می‌توانیم جدول دوطرفه تعداد افراد در جمعیت اولیه در برابر تعداد آنها در کلاسترها را بدست آوریم:

```
> table(pop(x), grp$grp)
```

	1	2	3	4	5	6
P1	97	0	0	3	0	0
P2	0	0	99	1	0	0
P3	0	0	0	0	98	2
P4	0	100	0	0	0	0
P5	1	2	0	2	0	95
P6	0	0	0	99	1	0

جدول فوق به عنوان مثال نشان می‌دهد که ۹۷ نفر از افراد جمعیت اول (P1) در کلاستر اول قرار گرفته‌اند و یا تمام افراد جمعیت چهارم (P4) در کلاستر دوم واقع شده‌اند. می‌توان با استفاده از تابع table.value، مندرجات جدول فوق را بصورت نموداری نمایش داد (شکل ۷-۲۰):

```
> table.value(table(pop(x), grp$grp), col.lab=paste("post", 1:6), row.lab=paste("prev", 1:6))
```



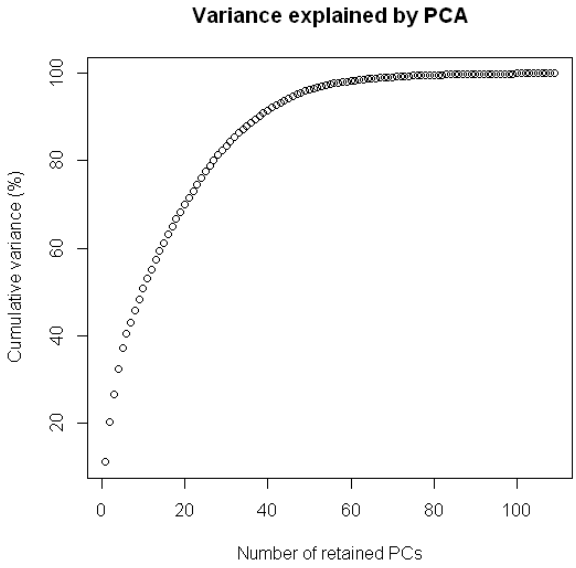
شکل ۷-۲۰ – نمایش تطبیقی عضویت افراد داده‌های a از مجموعه dapcIllus در جمعیت‌های از قبل موجود (prev) و کلاسترهای تشکیل شده (post) به روش K means

تجزیه تشخیصی مؤلفه‌های اصلی (DAPC)

تجزیه تشخیصی مؤلفه‌های اصلی[۲۴] (DAPC) یکی از روش‌های کارآمد در توصیف کلاسترهای ژنتیکی، می‌باشد. در این تکنیک، مجموعه‌ای از متغیرهای ساختگی به نام توابع تشخیصی براساس ترکیبات خطی از متغیرهای اولیه (آلل‌ها)، ایجاد می‌شود به نحوی که حداکثر اختلافات بین‌گروهی و حداقل تفاوت‌های درون‌گروهی را شامل شوند. براساس این روش، احتمال عضویت افراد در گروه‌های (کلاسترهای) متفاوت برآورد می‌شود که می‌تواند به عنوان معیاری از میزان شباهت (نزدیکی) افراد به گروه‌های مختلف، محسوب شود. این مقادیرِ مربوط به احتمالِ عضویت، می‌تواند شاهدی از میزان تفکیک و تمایز گروه‌ها از یکدیگر نیز باشد. به عبارت دیگر چنانچه گروه‌ها، تفکیک و تمایز واضحی از یکدیگر نداشته باشند، احتمال عضویت افراد در همه آنها تقریبا یکسان خواهد بود. از طرف دیگر، چنانچه گروه‌های متمایزی تشکیل شود، ولی برخی افراد را نتوان با احتمال بالا به یک گروه خاص منتسب کرد، این موضوع می‌تواند شاهدی از اختلاط موارد مربوطه باشد. نکته مهمی که باید توجه داشت اینکه برای انجام تجزیه DAPC، الزاماً بایستی از قبل، یک نوع گروه‌بندی در مجموعه داده‌ها وجود داشته باشد، یا مثلاً به روشی مانند گروه‌بندی K means، ایجاد شده باشد. دراینصورت تجزیه DAPC برمبنایِ این گروه‌هایِ از پیش تشکیل شده، انجام می‌شود (Jombart *et al.*, 2010). در اینجا به عنوان مثال، از نتایج تجزیه کلاستر K means، بر روی داده‌های a از مجموعه dapcIllus (که در بخش قبل توضیح داده شد) استفاده می‌شود. از آنجا که روش DAPC مبتنی بر تجزیه ابتدایی مؤلفه‌های اصلی به منظور کاهش حجم داده‌ها و ایجاد تفکیک واضح‌تری بین گروه‌ها می‌باشد، لازم است که در مورد تعداد مؤلفه‌های اصلی که باید نگه داشته شوند، تصمیم‌گیری شود. بدین منظور با اجرای تابع dapc، از کاربر درخواست می‌شود که این تعداد را مشخص نماید:

```
> library(adegenet)
> data(dapcIllus)
> x <- dapcIllus$a

> x

> dapc1 <- dapc(x, grp$grp)
```

[24] Discriminant Analysis of Principal Components



شکل ۷-۲۱ - درصد واریانس تجمعی توجیه شده توسط مؤلفه‌های اصلی در داده‌های a از مجموعه dapcIllus در مرحله اول از تجزیه DAPC

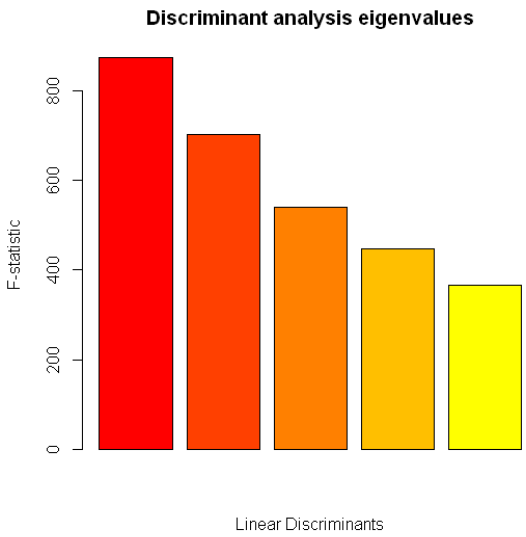
براساس نمودار شکل ۷-۲۱ می‌توان مشاهده کرد که از حدود ۴۰ مؤلفه اصلی به بعد، تغییر چندانی در افزایش واریانس تجمعی مشاهده نمی‌شود. لذا این عدد ۴۰ را می‌توان در پاسخ به سؤال مطرح شده از کاربر، وارد نمود.

```
Choose the number PCs to retain (>=1): 40
```

همانطور که گفته شد انتخاب تعداد مؤلفه‌های اصلی در این مرحله، نقش مؤثری در تفکیک و تمایز بین گروه‌ها در مراحل بعدی از این تجزیه دارد، لذا در این خصوص و نحوه بهینه سازی تعداد مؤلفه‌های اصلی اولیه انتخابی، در بخشی مجزا در ادامه مطالب توضیح داده خواهد شد.

در مرحله بعد، تعداد توابع تشخیصی که باید نگه داشته شوند، از کاربر سؤال می‌شود. در پاسخ به این سؤال، براساس نمودار ارائه شده مربوط به مقادیر ویژه (شکل ۷-۲۲)، می‌توان تعدادی را به دلخواه (مثلاً در اینجا پنج عدد) انتخاب کرد. در رابطه با نحوه بهینه سازی تعداد توابع تشخیصی که باید نگه داشته شوند نیز در ادامه، توضیح داده خواهد شد. لذا در این مرحله در پاسخ به سؤال مطرح شده از کاربر، عدد ۵ وارد می‌نماییم:

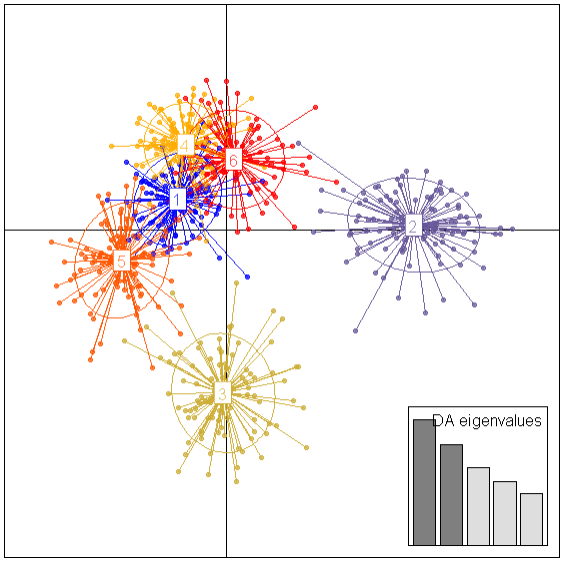
```
Choose the number discriminant functions to retain (>=1): 5
```



شکل ۷-۲۲ – نمودار ستونی مقادیر ویژه برای توابع تشخیصی مبتنی بر مؤلفه‌های اصلی در داده‌های a از مجموعه dapcIllus

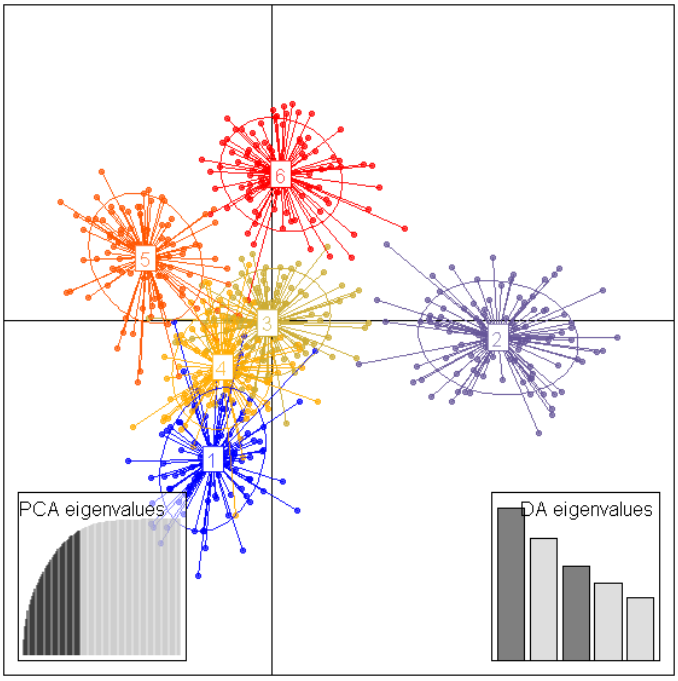
با استفاده از تابع scatter از پکیج ade4 بر روی خروجی تابع dapc، می‌توان نمودار توزیع افراد و جمعیت‌ها را براساس توابع تشخیصیِ انتخابی، ترسیم نمود. به عنوان مثال برای رسم نمودار پراکنش مبتنی بر توابع تشخیصی اول و دوم (شکل ۷-۲۳) و سپس اول و سوم (شکل ۷-۲۴)، دستورات زیر قابل اجرا می‌باشد:

```
> scatter(dapc1, xax=1, yax=2)
```



شکل ۷-۲۳ – تفکیک افراد در داده‌های a از مجموعه dapcIllus براساس توابع تشخیصی اول و دوم مبتنی بر مؤلفه‌های اصلی

> scatter(dapc1, xax=1, yax=3, scree.pca=T)



شکل ۷-۲۴ – تفکیک افراد در داده‌های a از مجموعه dapcIllus براساس توابع تشخیصی اول و سوم مبتنی بر مؤلفه‌های اصلی

خروجی تابع dapc، لیستی متشکل از اجزاء مختلف است که هریک حاوی اطلاعات مفیدی می‌باشند:

> dapc1

```
        ################################################
        # Discriminant Analysis of Principal Components #
        ################################################
    class: dapc
    $call: dapc.genind(x = x, pop = grp$grp)

    $n.pca: 40 first PCs of PCA used
    $n.da: 5 discriminant functions saved
    $var (proportion of conserved variance): 0.915

    $eig (eigenvalues): 874.1 703.2 541.5 447.9 365.3  vector    length content
```

1 $eig 5 eigenvalues

2 $grp 600 prior group assignment

3 $prior 6 prior group probabilities

4 $assign 600 posterior group assignment

5 $pca.cent 140 centring vector of PCA

6 $pca.norm 140 scaling vector of PCA

7 $pca.eig 109 eigenvalues of PCA

data.frame nrow ncol content

1 $tab 600 40 retained PCs of PCA

2 $means 6 40 group means

3 $loadings 40 5 loadings of variables

4 $ind.coord 600 5 coordinates of individuals (principal components)

5 $grp.coord 6 5 coordinates of groups

6 $posterior 600 6 posterior membership probabilities

7 $pca.loadings 140 40 PCA loadings of original variables

8 $var.contr 140 5 contribution of original variables

احتمال عضویت افراد در گروه‌های از پیش تعریف شده (کلاسترهای شش‌گانه، در اینجا) در جزء posterior، از خروجی تابع dapc، موجود می‌باشد:

> round(head(dapc1$posterior, 15),3)

	1	2	3	4	5	6
1	1.000	0.000	0.000	0.000	0.000	0.000
2	1.000	0.000	0.000	0.000	0.000	0.000
3	1.000	0.000	0.000	0.000	0.000	0.000
4	0.016	0.000	0.000	0.984	0.000	0.000
5	1.000	0.000	0.000	0.000	0.000	0.000
6	1.000	0.000	0.000	0.000	0.000	0.000
7	1.000	0.000	0.000	0.000	0.000	0.000
8	1.000	0.000	0.000	0.000	0.000	0.000
9	1.000	0.000	0.000	0.000	0.000	0.000
10	1.000	0.000	0.000	0.000	0.000	0.000
11	1.000	0.000	0.000	0.000	0.000	0.000
12	1.000	0.000	0.000	0.000	0.000	0.000
13	1.000	0.000	0.000	0.000	0.000	0.000
14	0.946	0.000	0.000	0.026	0.000	0.028
15	1.000	0.000	0.000	0.000	0.000	0.000

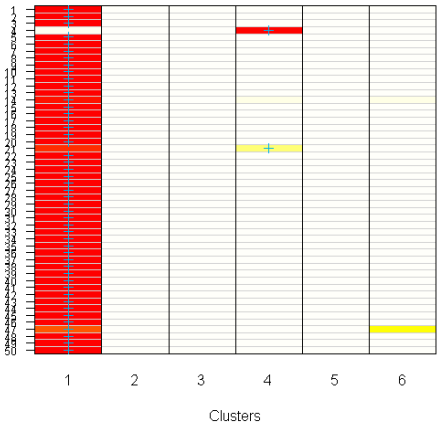
نتایج فوق به عنوان مثال نشان می‌دهد که افراد اول تا سوم به احتمال ۱۰۰ درصد به کلاستر اول تعلق دارند و فرد چهارم به احتمال ۱/۶ درصد به کلاستر اول و به احتمال ۹۸/۴ درصد به کلاستر چهارم متعلق می‌باشد. اطلاعات فوق را می‌توان با استفاده از تابع assignplot، بصورت نمودار نمایش داد (شکل ۷-۲۵):

```
> assignplot(dapc1, subset=1:50)
```

در دستور فوق آرگومان subset، نمودارِ نمایش را به افراد دلخواه محدود می‌کند که در اینجا به ۵۰ فرد اول محدود شده است. در این نمودار (شکل ۷-۲۵) برای احتمال عضویت یک (یا صد در صد) رنگ قرمز و برای احتمال عضویت صفر، رنگ سفید در نظر گرفته شد و سایر احتمالات، طیفی بین این دو رنگ است. علامت بعلاوه (+) آبی رنگ، گروه قبلی افراد (یعنی قبل از انجام تجزیه DAPC) را نشان می‌دهد. بنابراین امکان دارد فردی بعد از انجام DAPC، به احتمال زیادتری به گروه دیگری به غیر از آنچه که قبلاً در آن واقع بوده، منتسب شود. مثلاً فرد ۲۱، قبل از انجام DAPC، در گروه چهارم واقع بوده است، ولی بعد از انجام DAPC، مشخص شده که احتمال تعلق این فرد به گروه چهارم ضعیف (۰/۱۳۶) است و با احتمال قوی‌تری (۰/۸۶۴) به گروه اول قابل انتساب می‌باشد. مقادیر احتمال انتساب فرد ۲۱ به گروه‌های مختلف با دستور زیر قابل مشاهده است:

```
> round(dapc1$posterior[21,, drop=FALSE],3 )
```

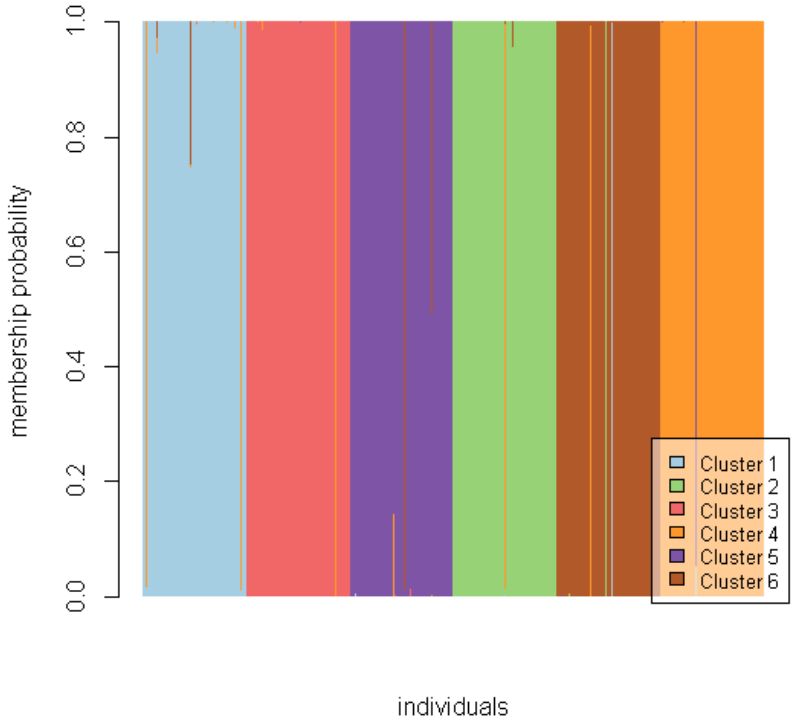
	1	2	3	4	5	6
21	0.864	0.000	0.000	0.136	0.000	0.000



شکل ۷-۲۵ - نمودار احتمال انتساب ۵۰ فرد اولِ داده‌های a از مجموعه dapcIllus به گروه‌های اولیه (تشکیل شده از قبل به روش K means) پس از انجام تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی (برای توضیح در مورد علایم و رنگ‌ها به متن مراجعه نمایید).

همچنین می‌توان احتمال انتساب افراد به گروه‌های مختلف را به شیوه‌ای مشابه با نرم‌افزار STRUCTUTE نمایش داد (شکل ۷-۲۶). این کار با دستور compoplot از پکیج adegenet قابل انجام می‌باشد:

```
compoplot(dapc1, posi="bottomright", txt.leg=paste("Cluster", 1:6), lab="", ncol=1, xlab="individuals", col=funky(6))
```

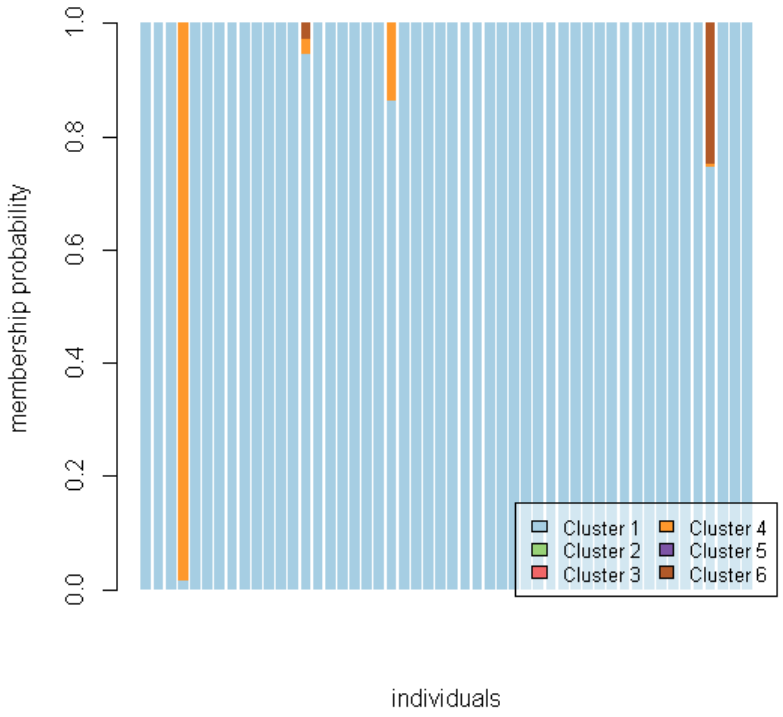


شکل ۷-۲۶ - نمودار احتمال عضویت افراد داده‌های a از مجموعه dapcIllus به گروه‌های اولیه (تشکیل شده از قبل به روش K means) پس از انجام تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی

در تابع فوق آرگومان‌های posi، txt.leg و ncol، مربوط به برچسب شرح علائم می‌باشند. آرگومان posi، محل قرار گرفتن برچسب شرح علائم، آرگومان txt.leg، چگونگی درج عناوین کلاسترها در درون برچسب شرح علائم و آرگومان ncol، تعداد ستون‌های عناوین در درون برچسب شرح علائم را مشخص می‌کند. همچنین آرگومان lab=""، سبب حذف شناسه افراد از زیر محور افقی می‌شود (از آنجا که تعداد افراد زیاد است، عدم درج این آرگومان سبب تراکم زیاد و روی هم افتادن اسامی و ناخوانا شدن آنها می‌شود).

در این تابع نیز آرگومان subset، نمودارِ نمایش احتمالِ انتساب به گروه‌ها را، به افراد دلخواه محدود می‌کند (شکل ۷-۲۷) مثلاً با محدود کردن نمودار به ۵۰ فرد اول خواهیم داشت:

```
> compoplot(dapc1, subset=1:50, posi="bottomright", txt.leg=paste("Cluster", 1:6), lab="", ncol=2, xlab="individuals", col=funky(6))
```



شکل ۷-۲۷ - نمودار احتمال عضویت ۵۰ فرد اولِ داده‌های a از مجموعه dapcIllus به گروه‌های اولیه (از قبل تشکیل شده به روش K means) پس از انجام تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی

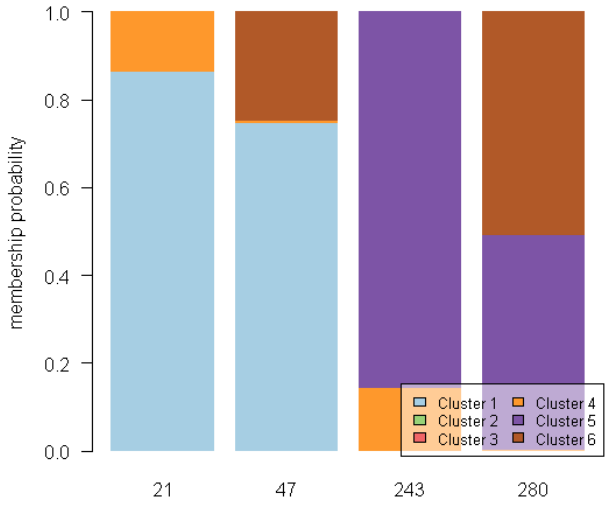
همچنین می‌توان به منظور بررسی دقیق‌تر، نمودار را به افراد اختلاط یافته محدود کرد. مثلاً با تابع which می‌توان افرادی که با احتمال کمتر از ۹۰ درصد، به کلاستر خاصی منتسب شده‌اند را، در متغیری (temp)، مجزا نمود و با استفاده از این متغیر در آرگومان subset، نمودار را صرفاً به نمایش احتمالات این افراد اختصاص داد (شکل ۷-۲۸):

```
> temp <-which(apply(dapc1$posterior,1, function(e) all(e<0.9)))
> temp
```

21 47 243 280

21 47 243 280

> compoplot(dapc1, subset=temp, posi="bottomright", txt.leg=paste("Cluster", 1:6), ncol=2, col=funky(6), las=1)



شکل ۷-۲۸ - نمودار احتمال عضویت افرادِ اختلاط یافته‌ی زیرمجموعه‌ی a از داده‌های dapcIllus به گروه‌های اولیه (تشکیل شده از قبل به روش K means) پس از انجام تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی

مشابه با روش تجزیه به مؤلفه‌های اصلی، در تجزیه DAPC نیز می‌توان آلل‌هایی که براساس توابع تشخیصی، بیشترین تأثیر در تفکیک جمعیت‌ها از یکدیگر دارند را، شناسایی نمود. نحوه انجام این عمل با استفاده از مجموعه داده H3N2 به عنوان مثال، شرح داده می‌شود. این مجموعه شامل داده‌های تعداد صد و بیست و پنج SNP، واقع در قطعه هماگلوتنین از ۱۹۰۳ سوش ویروس آنفولانزای فصلی است:

> library(adegenet)

> data(H3N2)

> H3N2

/// GENIND OBJECT /////////

// 1,903 individuals; 125 loci; 334 alleles; size: 3.1 Mb

// Basic content

@tab: 1903 x 334 matrix of allele counts

@loc.n.all: number of alleles per locus (range: 2-4)

@loc.fac: locus factor for the 334 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 1-1)

@type: codom

@call: .local(x = x, i = i, j = j, drop = drop)

// Optional content

@other: a list containing: x xy epid

از آنجا که در مجموعه داده H3N2، جمعیت‌های انتسابی افراد مشخص نشده است در گام اول به این امر اقدام می‌کنیم. در اینجا منظور از جمعیت، نمونه‌های جمع‌آوری شده از ویروس در سال‌های مختلف اپیدمی (۲۰۰۱ الی ۲۰۰۶) است. این داده‌ها در جزء epid از بخش other موجود است:

```
> head(H3N2$other$epid)
```

[1] 2001 2001 2001 2001 2001 2001

```
> tail(H3N2$other$epid)
```

[1] 2006 2006 2005 2005 2006 2006

همانطور که قبلاً شرح داده شد تابع pop، هم برای نمایش جمعیت‌های انتسابی افراد بکار می‌رود و هم به منظور انتساب کردن جمعیت‌ها به افراد، قابل کاربرد می‌باشد:

```
> pop(H3N2) <- H3N2$other$epid
> head(pop(H3N2))
```

[1] 2001 2001 2001 2001 2001 2001

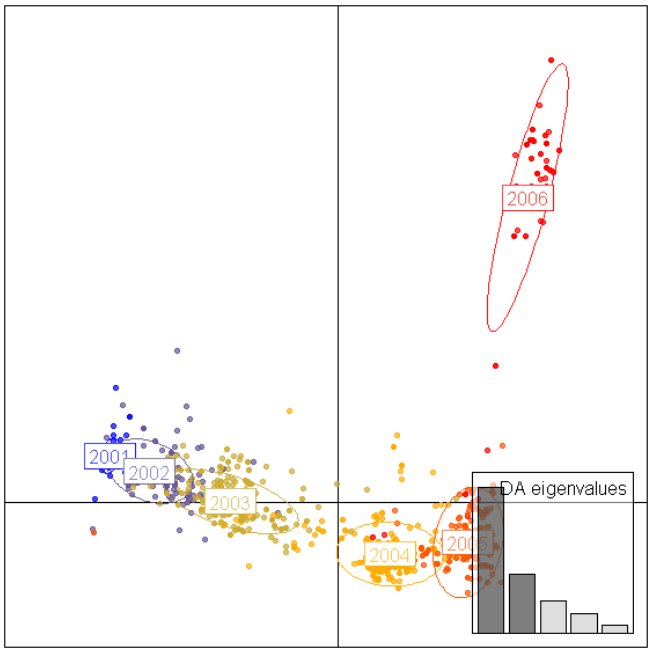
Levels: 2001 2002 2003 2004 2005 2006

اکنون تجزیه DAPC را با درنظر گرفتن تعداد ۳۰ مؤلفه اصلی و نگه داشتن ۱۰ تابع تشخیصی، انجام می‌دهیم و نمودار پراکنش افراد و جمعیت‌ها را ترسیم می‌نماییم (شکل ۷-۲۹):

```
> dapc.flu<- dapc(H3N2, n.pca=30,n.da=10)
```

```
> scatter(dapc.flu, cstar=0)
```

توجه داشته باشید که در دستور فوق به منظور وضوح بیشتر نمودار، خطوط متصل کننده نقاط مربوط به افراد به مرکز جمعیت‌ها، با استفاده از آرگومان cstar=0 حذف شده است.



شکل ۷-۲۹ – تفکیک افراد مجموعه داده H3N2 براساس توابع اول و دوم از تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی

همانطور که در نمودار فوق مشاهده می‌شود، نمودار افقی، جمعیت‌های ویروس را برحسب سال‌های اپیدمی بخوبی از یکدیگر مجزا نموده است و لذا تابع تشخیصی اول، نشان‌دهنده تکامل ویروس H3N2 در طی زمان می‌باشد. اما محور عمودی نقش ویژه‌ای در تمایز جمعیت سال ۲۰۰۶ از سایر جمعیت‌ها داشته است. بنابراین شناسایی آلل‌هایی که در این تمایز بیشترین سهم را داشته‌اند، جالب خواهد بود. این اطلاعات در جزء var.contr از خروجی تابع dapc موجود است:

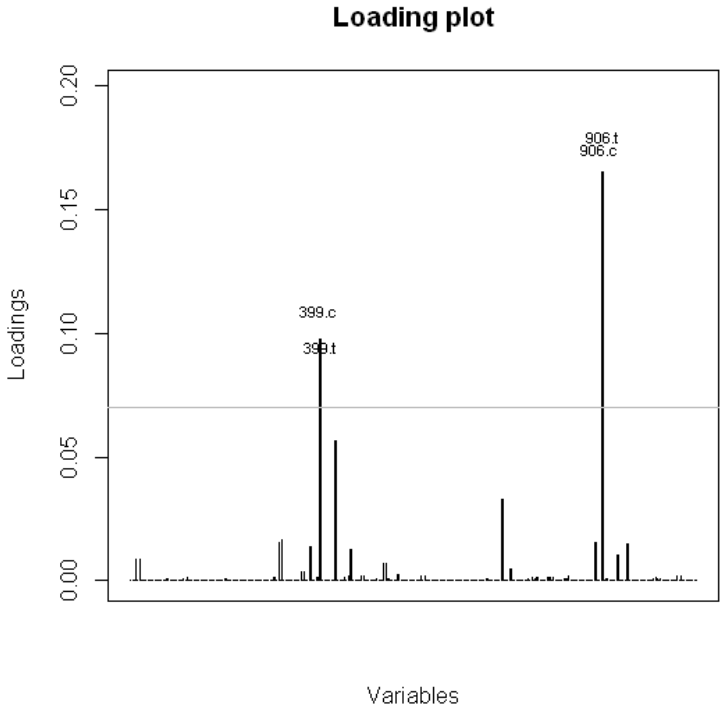
```
> head(dapc.flu$var.contr)
```

	LD1	LD2	LD3	LD4	LD5
6.a	2.043722e-02	1.628288e-04	2.787775e-02	1.187327e-03	4.251438e-02
6.c	6.276535e-06	4.928553e-07	2.775708e-05	2.400245e-06	2.724773e-05
6.g	1.972719e-02	1.454050e-04	2.614619e-02	1.082959e-03	4.038903e-02
17.a	1.553231e-03	8.587292e-03	5.043203e-03	1.291304e-03	3.151817e-04

```
17.g 1.440502e-05 8.960161e-08 3.512841e-05 5.917724e-05 1.069455e-04
17.t 1.268475e-03 8.642859e-03 5.920138e-03 7.976132e-04 5.493644e-05
> class(dapc.flu$var.contr)
[1] "matrix"
```

با استفاده از تابع loadingplot در پکیج adegenet، ضرایب مندرج در ماتریس فوق، به صورت نمودار قابل نمایش است (شکل ۷-۳۰):

```
> set.seed(4)
> contrib<- loadingplot(dapc.flu$var.contr, axis=2, thres=.07, lab.jitter=1)
> contrib
$threshold
[1] 0.07

$var.names
[1] "399.c" "399.t" "906.c" "906.t"

$var.idx
399.c 399.t 906.c 906.t
  112   113   278   279

$var.values
    399.c         399.t         906.c     906.t
0.09750789 0.09750789 0.16524788 0.16524788
```



شکل ۷-۳۰ – شناسایی SNPهای دارای بیشترین نقش در دومین تابع از تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی در مجموعه داده H3N2

در تابع فوق با استفاده از آرگومان axis=2، فقط ضرایب تابع تشخیصی دوم (مربوط به محور عمودی) مدنظر قرار گرفته است. همچنین با استفاده از آرگومان lab.jitter محل قرار گرفتن اسامی در نمودار به منظور احتراز از روی هم افتادن آنها، تصادفی شده و دستور set.seed برای تکرارپذیری نتایج، بکار رفته است.

همانطور که هم در نمودار و هم در خروجی تابع loadingplot، مشخص است، آلل‌های SNP در لوکوس‌های 906 و 399 بیشترین نقش را در تابع تشخیصی دوم داشته‌اند. اکنون می‌توان تعییر فراوانی آلل‌های این SNPها را در سال‌های مختلف بررسی کرد و با مقایسه آنها پی برد که چه تغییری سبب تمایزِ جمعیتِ سال ۲۰۰۶، از سایر جمعیت‌ها شده است. برای این کار، ابتدا لوکوس 906 را در قالب شیء genind مجزا، ذخیره می‌نماییم:

```
> H3N2[loc=c("906")]
        /// GENIND OBJECT /////////
         // 1,903 individuals; 1 locus; 2 alleles; size: 630.8 Kb
         // Basic content
```

@tab: 1903 x 2 matrix of allele counts

@loc.n.all: number of alleles per locus (range: 2-2)

@loc.fac: locus factor for the 2 columns of @tab

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 1-1)

@type: codom

@call: .local(x = x, i = i, j = j, loc = ..1, drop = drop)

// Optional content

@pop: population of each individual (group size range: 107-545)

@other: a list containing: x xy epid

از دستور genind2genpop برای تبدیل شیء genind به شیء genpop استفاده می‌کنیم تا بتوانیم تغییر فراوانی آلل‌های لوکوس 906 طی سال‌های مختلف را در جمعیت‌ها (بجای افراد) بررسی نماییم:

```
> loc.906<-genind2genpop(H3N2[loc=c("906")])
> loc.906
/// GENPOP OBJECT /////////
 // 6 populations; 1 locus; 2 alleles; size: 518.9 Kb
 // Basic content
   @tab:  6 x 2 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 2-2)
   @loc.fac: locus factor for the 2 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 1-1)
   @type:  codom
   @call: genind2genpop(x = H3N2[loc = c("906")])
 // Optional content
   @other: a list containing: x  xy  epid
> tab(loc.906)
      906.c 906.t
2001      0   100
2002      0   207
2003      0   407
```

2004	0	499
2005	1	463
2006	69	43

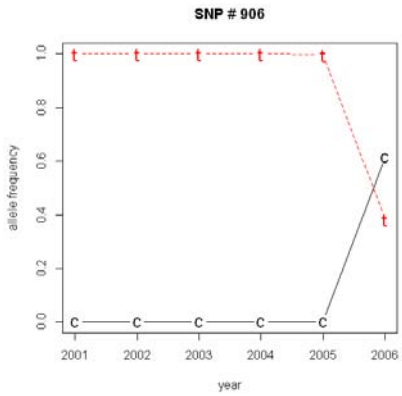
با استفاده از آرگومان freq=T در تابع tab، می‌توان فراوانی نسبی آلل‌ها را (بجای فراوانی مطلق) بدست آورد:

```
> freq.906<-tab(loc.906, freq=T)
> freq.906
```

	906.c	906.t
2001	0.000000	1.000000
2002	0.000000	1.000000
2003	0.000000	1.000000
2004	0.000000	1.000000
2005	0.0021552	0.9978448
2006	0.6160714	0.3839286

همانطور که در جدول فوق مشخص است، فراونی نسبی آلل‌های c و t در لوکوس 906 تا سال ۲۰۰۴ به ترتیب صفر و یک بوده است، ولی از سال ۲۰۰۵ و به بعد، فراوانی نسبی آلل t رو به کاهش و فراوانی نسبی آلل c، رو به افزایش گذاشته است. این تغییرات در فراوانی آلل‌ها را می‌توان با استفاده از تابع matplot نمایش داد (شکل ۷-۳۱):

```
> matplot(freq.906, pch=c("c","t"), type="b", xlab="year", ylab="allele frequency", xaxt="n", cex=1.5, main="SNP # 906")
> axis(side=1, at=1:6, lab=2001:2006)
```



شکل ۷-۳۱ - روند تغییر فراونی نسبی آلل‌های c و t در لوکوس 906 واقع در قطعه هماگلوتنین از ۱۹۰۳ سوش ویروس آنفولانزای فصلی طی سال‌های ۲۰۰۱ تا ۲۰۰۶ در مجموعه داده H3N2

انتخاب تعداد بهینه مؤلفه‌های اصلی برای تجزیه DAPC

تعداد مؤلفه‌های اصلی که برای تجزیه DAPC انتخاب می‌شود تأثیر مهمی در نتایج این تجزیه دارد، لذا دقت در انتخاب تعداد بهینه مؤلفه اصلی از اهمیت برخوردار است. چنانچه تعداد مؤلفه اصلیِ نگه داشته شده کم باشد، مقدار زیادی از داده‌های ژنتیکی از دست خواهد رفت. از سوی دیگر اگر تعداد زیادی مؤلفه اصلی (یا همه آنها) نگهداری شود، گروه‌ها بخوبی از یکدیگر تمایز نمی‌یابند یعنی افراد بجای اینکه تفکیک شده و با احتمال بالایی به گروه‌های خاصی منتسب شوند، همگی با احتمال مشابه به همه گروه‌ها منتسب می‌شوند و بدین ترتیب تجزیه نخواهد توانست هیچ‌گونه ساختاری را در داده‌های موجود تشخیص دهد. بنابراین معیاری لازم است تا توسط آن مشخص شود که بهترین تعداد مؤلفه اصلی برای انجام تجزیه DAPC چیست. در این رابطه کمیتی بنام امتیاز α[۲۵] تعریف شده است که عبارتست از تفاوت بین نسبت انتساب‌های موفق حاصل از انجام تجزیه DAPC مبتنی بر گروه‌های اصلی با نسبت انتساب‌های موفقی که بطور تصادفی انجام شده‌اند. به عبارت دیگر در این روش، گروه‌های تصادفی از افراد (بجای گروه‌های از قبل مشخص) تعریف می‌شود و تجزیه DAPC روی این داده‌ها انجام می‌گیرد و درصد انتساب موفق افراد به گروه‌ها محاسبه می‌شود و با درصد انتساب موفق افراد به گروه‌ها در تجزیه DAPC اصلی (یعنی تجزیه‌ای که مبتنی بر گروه‌هایِ از قبل مشخص، است) مقایسه و تفاوت آنها محاسبه می‌شود. بدین ترتیب هرچه میزان امتیاز α بیشتر باشد، نشان‌دهنده غیرتصادفی بودن انتساب‌های موفق در تجزیه اصلی است. لذا از این معیار می‌توان برای یافتنِ بهترین تعدادِ مؤلفه اصلیِ انتخابی، استفاده نمود. به عنوان مثال تجزیه DAPC را بر روی مجموعه داده microbov با در نظر گرفتن ۱۰ مؤلفه اصلی (n.pca=10) و نگه داشتن ۱۰۰ تابع تشخیصی (n.da=100) انجام می‌دهیم و با اِعمال تابع a.score بر روی خروجی تابع dapc، کمیت امتیاز α را محاسبه می‌کنیم:

```
> library(adegenet)
> data(microbov)
> dapc10 <- dapc(microbov, n.da=100, n.pca=10)
> temp10<- a.score(dapc10)
```

خروجی تابع a.score لیستی متشکل از اجزاء مختلف است:

```
> names(temp10)
[1] "tab"       "pop.score" "mean"
> temp10$tab[1:5,1:5]
```

[25] α-Score

	Borgou	Zebu	Lagunaire	NDama	Somba
sim.1	0.66	0.64	0.843137	0.5	0.64
sim.2	0.62	0.82	0.843137	0.5	0.82
sim.3	0.56	0.68	0.784314	0.566667	0.74
sim.4	0.54	0.8	0.843137	0.5	0.58
sim.5	0.56	0.88	0.921569	0.566667	0.68

```
> temp10$pop.score
```

Borgou	Zebu	Lagunaire	NDama	Somba
0.612	0.802	0.8607843	0.5233333	0.684
Aubrac	Bazadais	BlondeAquitaine	BretPieNoire	Charolais
0.492	0.8723404	0.3327869	0.5225806	0.6018182
Gascon	Limousin	MaineAnjou	Montbeliard	Salers
0.682	0.4	0.8183673	0.67	0.754

جزء tab شامل مقدار امتیاز α در شبیه‌سازی‌های مختلف به تفکیک جمعیت‌ها است و جزء pop.score میانگین امتیاز α در شبیه‌سازی‌های مختلف به تفکیک جمعیت‌ها می‌باشد. می‌توان صحت این موضوع را مثلاً برای جمعیت اول (Borgou)، با دستورات زیر بررسی و تأیید نمود:

```
> temp10$tab[,1]
```

sim.1	sim.2	sim.3	sim.4	sim.5	sim.6	sim.7	sim.8	sim.9	sim.10
0.66	0.62	0.56	0.54	0.56	0.66	0.66	0.58	0.7	0.58

```
> mean(temp10$tab[,1])
    [1] 0.612
> temp10$pop.score[1]
    Borgou
     0.612
```

جزء mean شامل میانگینِ تمام امتیازهای α محاسبه شده در تمام شبیه‌سازی‌هایِ جمعیت‌ها است.

```
> temp10$mean
    [1] 0.6418674
> mean(temp10$tab)
    [1] 0.6418674
```

نتایج فوق نشان می‌دهد که نسبت انتساب موفق افراد به جمعیت‌ها در تجزیه DAPC اصلی، حدود ۶۴ درصد بیشتر از تجزیه DAPC مبتنی بر گروه‌بندیِ تصادفیِ افراد است.

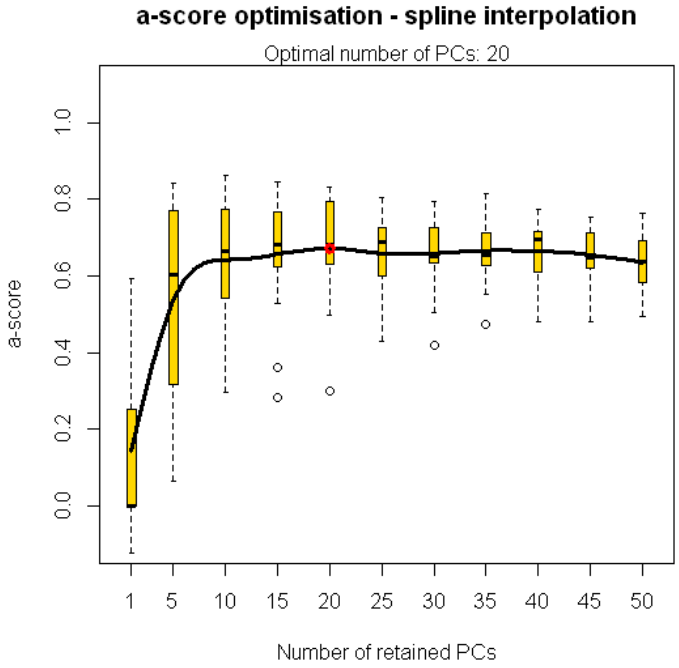
اکنون به منظور مقایسه، تجزیه‌های متوالی DAPC را بر روی مجموعه داده microbov با در نظر گرفتن ۵، ۲۰ و ۵۰ مؤلفه اصلی و نگه داشتن ۱۰۰ تابع تشخیصی (n.da=100) انجام می‌دهیم و با اِعمال تابع a.score بر روی خروجی تابع dapc، کمیت امتیاز α را محاسبه می‌کنیم:

```
> dapc5 <- dapc(microbov, n.da=100, n.pca=5)
> temp5 <- a.score(dapc5)
> temp5$mean
    [1] 0.5408815
> dapc20 <- dapc(microbov, n.da=100, n.pca=20)
> temp20 <- a.score(dapc20)
> temp20$mean
    [1] 0.6724406
> dapc50 <- dapc(microbov, n.da=100, n.pca=50)
> temp50 <- a.score(dapc50)
> temp50$mean
    [1] 0.6465957
```

(توجه داشته باشید از آنجا که روش فوق برمبنای رویه تصادفی است، اعداد بدست آمده برای شما ممکن است با اعداد فوق، یکسان نباشد، ولی نتیجه کلی مشابه خواهد بود).

مشاهده می‌شود که براساس کمیّت امتیاز α، کارایی تجزیه DAPC با کاهش تعداد مؤلفه‌های اصلی از ۱۰ به ۵ کاسته شده است (امتیاز α از ۰/۶۴ به ۰/۵۴ کاهش یافته) اما با افزایش تعداد مؤلفه‌های اصلی از ۱۰ به ۲۰ بر کارایی تجزیه DAPC افزوده شده (امتیاز α از ۰/۶۴ به ۰/۶۷ افزایش یافته) است، اما مجدداً با افزایش تعداد مؤلفه‌های اصلی از ۲۰ به ۵۰، از کارایی تجزیه DAPC (نسبت به حالت ۲۰ مؤلفه اصلی) کاسته شده است. بنابراین در محدوده‌ای که تجزیه‌های DAPC را بطور متوالی انجام دادیم، احتمالاً بهترین حالت در حدود ۲۰ مؤلفه اصلی می‌تواند باشد. به منظور اینکه نیازی به انجام این تجزیه‌های مکرر نباشد، تابعی به نام optim.a.score تعریف شده که بر روی خروجی تابع dapc قابل اعمال است و بهترین تعداد مؤلفه‌های اصلی انتخابی را برای انجام dapc، شناسایی کرده و با رسم نمودار نشان می‌دهد (شکل ۷-۳۳). برای کار با این تابع، ابتدا در تابع dapc، آرگومان n.pca را برای تعداد مؤلفه‌های اصلی بیشتر (در مثال فوق بیشترین تعداد مؤلفه‌های اصلی ۵۰ بود) در نظر بگیرید، تا امتیاز α، برای دامنه یک تا آن تعداد مؤلفه اصلی (در اینجا یک تا ۵۰ مؤلفه اصلی) محاسبه شود:

```
> dapc50 <- dapc(microbov, n.da=100, n.pca=50)
> opti <- optim.a.score(dapc50)
```



شکل ۷-۳۲ – مقادیر امتیاز α برحسب تعداد مختلف مؤلفه‌های اصلی انتخاب شده برای انجام تجزیه DAPC در مجموعه داده microbov

همانطور که مشاهده می‌شود نتایج فوق با پیش‌بینی ما مطابقت دارد و لذا تجزیه DAPC با انتخاب ۲۰ مؤلفه اصلی دارای بیشترین کارایی خواهد بود. توجه داشته باشید از آنجا که این رویه براساس شبیه‌سازی و تصادفی‌سازی می‌باشد، ممکن است انجام مجددِ آن، دقیقاً به همین تعداد ۲۰ مؤلفه اصلی منتهی نشود، ولی یقیناً در همین حدود خواهد بود.

فصل هشتم- بررسی سطوح ساختاری جمعیت

آزمون ساختاریافتگی جمعیت

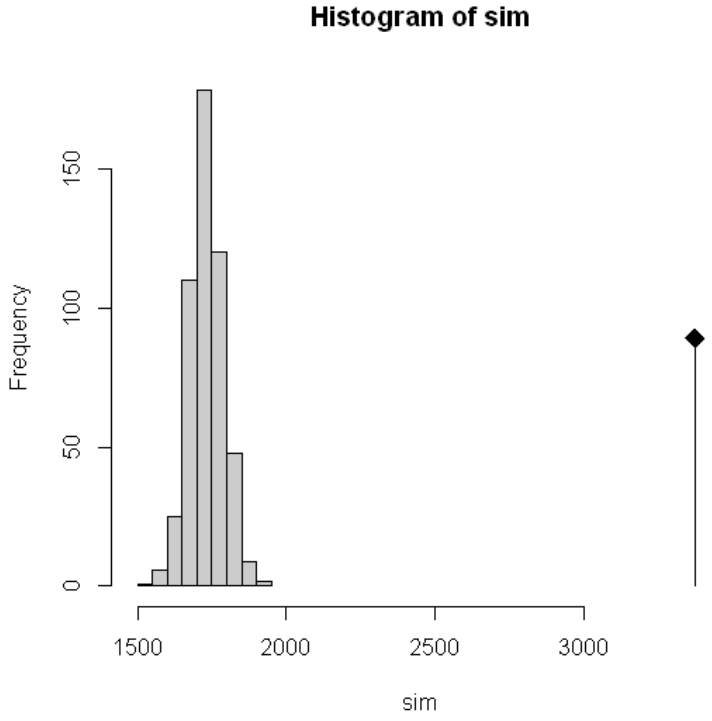
ساختاریافتگی ژنتیکی جمعیت را می‌توان با استفاده از آماره G مربوط به نسبت درستنمایی (Goudet *et.al.*, 1996) آزمون نمود. برای این منظور تابع gstat.randtest در پکیج hierfstat قابل استفاده می‌باشد. توسط این تابع می‌توان آزمون ساختاریافتگی ژنتیکی را با استفاده از آماره G و به روش مونت کارلو در سطوح مختلف، بصورت کلی (method = global)، یا سلسله مراتبی (method = within یا method = between) انجام داد.:

gstat.randtest(x, pop = NULL, method = c("global", "within", "between"), sup.pop = NULL, sub.pop = NULL, nsim = 499)

در تابع فوق، x یک شیء genind است. جمعیت انتسابی افراد را می‌توان با استفاده از آرگومان pop تعیین کرد و درصورت NULL بودن این آرگومان یا عدم ذکر آن، داده‌های موجود در جزء @pop مدنظر قرار خواهد گرفت. اولین آرگومان sup.pop مربوط به هر نوع گروه‌بندی، با مقیاسی بزرگتر از آنچه که در جزء pop از شیء genind تعریف شده، می‌باشد که در روش within بکار می‌رود. دومین آرگومان sup.pop مربوط به هر نوع گروه‌بندی، با مقیاس کوچکتر از آنچه که در جزء pop از شیء genind تعریف شده، می‌باشد و در روش between بکار می‌رود. آرگومان nsim نیز تعداد تکرارِ رویه‌ی شبیه‌سازی می‌باشد. به عنوان مثال آزمون ساختاریافتگی ژنتیکی در مجموعه داده nancycats بصورت زیر قابل انجام خواهد بود (شکل ۸-۱):

```
> library(hierfstat)
> library(adegenet)
```

```
>  data(nancycats)
> cats.G<- gstat.randtest(nancycats)
> cats.G
        Monte-Carlo test
        Call: gstat.randtest(x = nancycats)

        Observation: 3372.926

        Based on 499 replicates
        Simulated p-value: 0.002
        Alternative hypothesis: greater

          Std.Obs Expectation    Variance
        28.63571  1732.21329  3282.83461

>  plot(cats.G)
```



شکل ۸-۱- توزیع فراوانی مرجع برای آزمون ساختاریافتگی ژنتیکی در مجموعه داده nancycats به روش مونت کارلو برمبنای جمعیت‌های انتسابی افراد

نتایج آزمون و نمودار حاصل از شبیه‌سازی، وجود ساختاریافتگی بسیار معنی‌دار در داده‌های nancycats را برمبنای جمعیت‌های انتسابی افراد (که در جزء pop از این شیء مندرج است)، نشان می‌دهد.

درصورتی که گروه‌بندی فراتر از این جمعیت‌ها یا در مقیاس کوچکتر از آن وجود داشته باشد، یعنی جمعیت‌ها خود، در داخل گروه‌های بزرگتری قرار داشته باشند و یا اینکه افراد در درون خودِ جمعیت‌ها،

بصورت جزئی‌تری دسته بندی شده باشند، در آن صورت همانطور که اشاره شد می‌توان با استفاده از آرگومان‌های method = within و method = between، ساختار یافتگی ژنتیکی را آزمون نمود. در این رابطه به عنوان مثال می‌توان مجموعه داده gtrunchier را مورد بررسی قرار داد. این مجموعه از نوع قالب داده (data frame) و شامل داده‌های ژنوتیپی مربوط به شش نشانگر میکروستلایت در ۳۷۰ فرد از نوعی حلزون (*Galba truncatula*) می‌باشد که از مکان‌های (Locality) مختلف و در قالب چند گروه (Patch) از هر مکان جمع‌آوری شده‌اند:

```
> data(gtrunchier)
> class(gtrunchier)
        [1] "data.frame"
> head(gtrunchier)
```

	Locality	Patch	L21.V	L37.J	L20.B	L29.V	L36.B	L16.J
1	1	1	22	11	11	11	55	55
2	1	1	77	11	55	11	22	55
3	1	1	22	11	55	22	55	55
4	1	1	77	11	55	11	22	55
5	1	1	27	23	55	22	22	57
6	1	1	22	11	55	22	55	55

```
> tail(gtrunchier)
```

	Locality	Patch	L21.V	L37.J	L20.B	L29.V	L36.B	L16.J
365	6	28	55	33	56	23	56	12
366	6	29	55	33	55	22	66	22
367	6	29	25	33	56	33	66	22
368	6	29	55	33	55	33	33	11
369	6	29	22	23	55	55	66	12
370	6	29	55	33	56	33	36	12

از آنجا که تابع gstat.randtest بر روی اشیاء genind عمل می‌نماید، لازم است قالب داده gtrunchier را به شیء genind تبدیل نماییم. اما از آنجا که وجود نقطه (.) در شناسه‌ی ستون‌های قالب داده gtrunchier هنگام تبدیل آن به شیء genind، ایجاد خطا می‌کند، ابتدا نسبت به حذف آن عمل می‌نماییم:

```
> colnames(gtrunchier)
        [1] "Locality" "Patch"   "L21.V"   "L37.J"   "L20.B"   "L29.V"   "L36.B"
        [8] "L16.J"
> colnames(gtrunchier)[-c(1,2)]=c("L21V", "L37J", "L20B", "L29V", "L36B", "L16J")
> colnames(gtrunchier)
        [1] "Locality" "Patch"   "L21V"    "L37J"    "L20B"    "L29V"    "L36B"
        [8] "L16J"
> x <- df2genind(gtrunchier[,-c(1,2)],ploidy=2, ncode=1 ,pop=gtrunchier$Patch)
> x
        /// GENIND OBJECT /////////
         // 370 individuals; 6 loci; 34 alleles; size: 72.4 Kb
```

```
        // Basic content
          @tab:  370 x 34 matrix of allele counts
          @loc.n.all: number of alleles per locus (range: 3-7)
          @loc.fac: locus factor for the 34 columns of @tab
          @all.names: list of allele names for each locus
          @ploidy: ploidy of each individual  (range: 2-2)
          @type:  codom
          @call: df2genind(X = gtrunchier[, -c(1, 2)], ncode = 1, pop = gtrunchier$Patch,
           ploidy = 2)

        // Optional content
          @pop: population of each individual (group size range: 5-15)
> tab(x)[1:10,1:7]
```

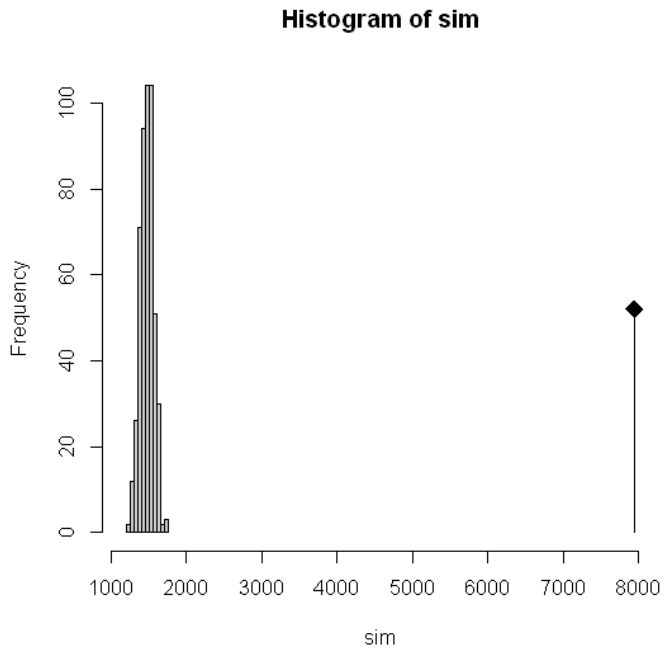
	L21V.2	L21V.7	L21V.3	L21V.6	L21V.4	L21V.5	L21V.1
1	2	0	0	0	0	0	0
2	0	2	0	0	0	0	0
3	2	0	0	0	0	0	0
4	0	2	0	0	0	0	0
5	1	1	0	0	0	0	0
6	2	0	0	0	0	0	0
7	2	0	0	0	0	0	0
8	0	2	0	0	0	0	0
9	2	0	0	0	0	0	0
10	0	2	0	0	0	0	0

```
> pop(x)
        [1] 1  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  2  2
        [26] 3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4
        ……
        [351] 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 29 29 29 29 29
        29 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 ... 29
> nPop(x)
        [1] 29
```

همانطور که مشاهد می‌شود ۲۹ گروه (Patch) در قالب داده اولیه gtrunchier، به عنوان pop در شیء genind جدید لحاظ گردیده‌اند. بنابراین دسته‌بندی مکان‌ها (Locality)، دارای مقیاسی بزرگتر از این pop می‌باشد، یعنی در اینحالت گروه‌های مندرج در pop، در درون مکان‌ها (Locality) قرار دارند. در اینجا اگر تابع بطور عمومی (یعنی بصورت gstat.randtest(x)) بکار گرفته شود، افتراق جمعیت صرفاً براساس گروه‌های pop (یعنی Patchها) به عنوان جمعیت‌ها (یعنی بدون درنظر گرفتن اینکه گروه‌ها در داخل مکان‌ها قرار دارند)، آزمون می‌شود (شکل ۸-۲):

```
> x.g<-gstat.randtest(x)
> x.g
        Monte-Carlo test
        Call: gstat.randtest(x = x)

        Observation: 7941.267
```

```
        Based on 499 replicates
        Simulated p-value: 0.002
        Alternative hypothesis: greater

          Std.Obs Expectation    Variance
          74.2149   1469.6112   7604.1126
> plot(x.g)
```



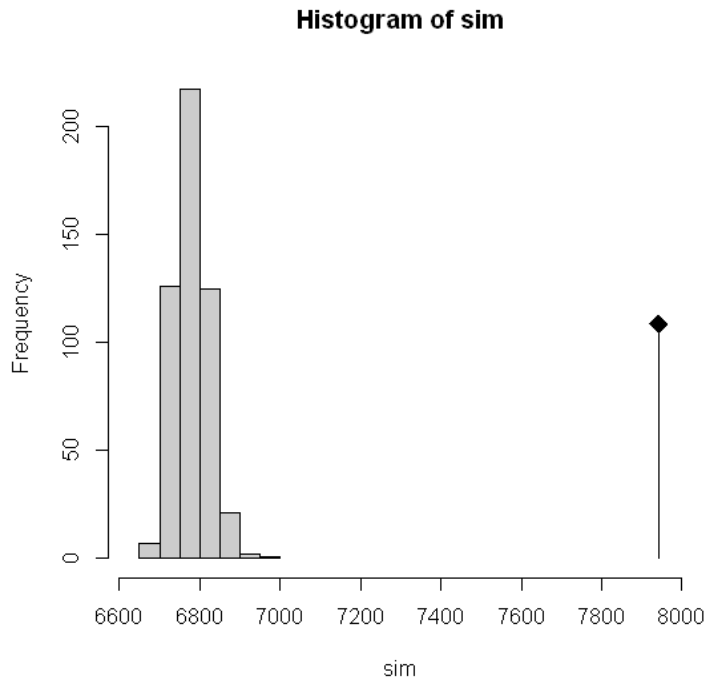
شکل ۸-۲- توزیع فراوانی مرجع برای آزمون ساختاریافتگی ژنتیکی در مجموعه داده gtrunchier به روش مونت کارلو برمبنای گروه‌های (Patch) مختلف بدون درنظر گرفتن مکان‌ها (Locality)

اما برای اینکه افتراق ژنتیکی گروه‌ها را با درنظر گرفتن مکان‌ها (Locality)، که دارای مقیاسی فراتر از جمعیت‌های pop هستند، آزمون نماییم باید از آرگومان "method="within و با تعریف Locality در آرگومان sup.pop، به شرح زیر استفاده کنیم. در اینحالت اثر مکان (Locality) ثابت نگه داشته می‌شود و تفاوت بین گروه‌ها (Patch) آزمون می‌شود (شکل ۸-۳).

توجه: گزینه within به قرار گرفتن گروه‌ها در درون مکان‌ها اشاره دارد.

```
> x.w<-gstat.randtest(x, method="within", sup.pop=gtrunchier$Locality)
> x.w
        Monte-Carlo test
        Call: gstat.randtest(x = x, method = "within", sup.pop = gtrunchier$Locality)

        Observation: 7941.267

        Based on 499 replicates
        Simulated p-value: 0.002
        Alternative hypothesis: greater
```

```
        Std.Obs Expectation    Variance
        27.6581   6778.9529   1766.0464
> plot(x.w)
```



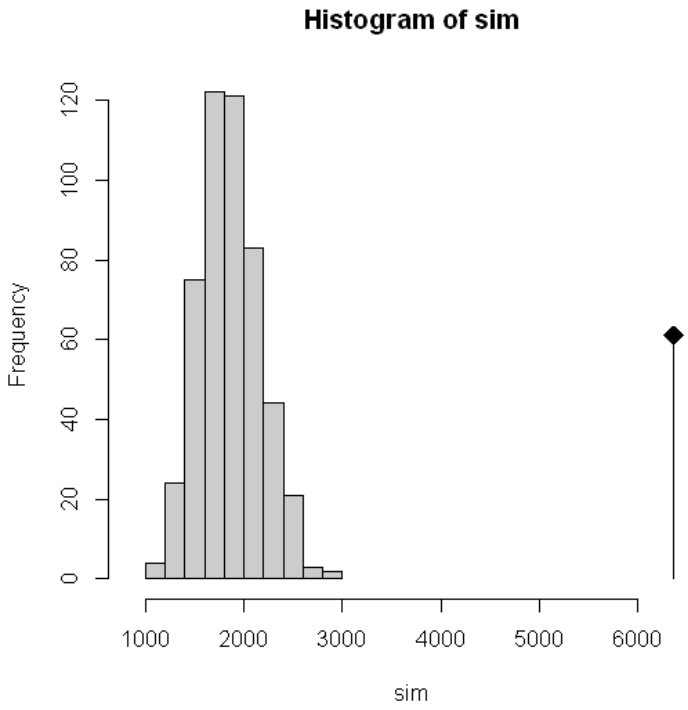
شکل ۸-۳- توزیع فراوانی مرجع برای آزمون ساختاریافتگی ژنتیکی در مجموعه داده gtrunchier به روش مونت کارلو برمبنای گروه‌های (Patch) مختلف با ثابت در نظر گرفتن اثر مکان (Locality)

توجه داشته باشید از آنجاکه pop را قبلاً (هنگام تبدیل داده‌های gtrunchier به شیء genind) تعریف نمودیم، لزومی به استفاده از آرگومان pop در دستور فوق نبود و خود تابع gstat.randtest آنها را از جزء pop@ از شیء genind بدست می‌آورد.

اکنون برای اینکه بتوانیم آزمون ساختاریافتگی ژنتیکی را براساس مکان‌ها و با درنظر گرفتن گروه‌ها (Patch) در درون مکان‌ها (Locality) انجام دهیم، لازم است که Locality را به عنوان pop لحاظ کرده و از آرگومان method="between" استفاده نماییم (شکل ۸-۴). در این حالت اثر گروه (Patch) ثابت نگه داشته می‌شود:

```
> x.b<-gstat.randtest(x, pop=gtrunchier$Locality, method="between",
sub.pop=gtrunchier$Patch)
> x.b
        Monte-Carlo test
        Call: gstat.randtest(x = x, pop = gtrunchier$Locality, method = "between",
          sub.pop = gtrunchier$Patch)

        Observation: 6370.574

        Based on 499 replicates
        Simulated p-value: 0.002
        Alternative hypothesis: greater
```

```
      Std.Obs Expectation    Variance
     14.56645  1858.55452 95947.64870
> plot(x.b)
```



شکل ۸-۴- توزیع فراوانی مرجع برای آزمون ساختاریافتگی ژنتیکی در مجموعه داده gtrunchier به روش مونت کارلو برمبنای مکان‌های (Locality) مختلف با ثابت در نظر گرفتن اثر گروه (Patch)

تجزیه واریانس مولکولی (AMOVA)

هنگامی که یک جمعیت به زیر جمعیت‌هایی تقسیم می‌شود، میزان هتروزیگوسیتی نسبت به جمعیت تفکیک نیافته، کاهش می‌یابد. اندازه این بخش‌های فرعی جمعیت از جمعیت اصلی کوچکتر است و از آنجا که فراوانی آللی در هر نسل نمونه‌ای از فراوانی آلل نسل قبلی است، اشتباه نمونه‌برداری بزرگتری نسبت به جمعیت اصلیِ تفکیک نیافته رخ خواهد داد. لذا پدیده رانده شدن ژنتیکی[۲۶] این بخش‌های فرعی را به سمت فراوانی‌های آللی متفاوتی سوق خواهد داد و تثبیت شدن آلل‌ها با نرخ سریعتری نسبت به جمعیت اصلیِ تفکیک نیافته، روی می‌دهد. در نتیجه وقتی در یک گونه، چندین زیرجمعیت به لحاظ جغرافیایی از یکدیگر تفکیک می‌شوند، واریانس ژنتیکی به دلیل افتراق ژنتیکی زیرجمعیت‌ها افزایش می‌یابد. با ظهور نشانگرهای مولکولی، برآورد میزان تفاوت‌های جهشی در ژن‌های مختلف، علاوه بر بررسی فراوانی آنها، امکان‌پذیر شده

[26] Genetic drift

است. به منظور مطالعه تنوع مولکولی درون یک گونه، روش تجزیه سلسه مراتبی واریانس مولکولی[۲۷] (AMOVA) توسط اکسکوفیر (Excoffier *et al.*, 1992) معرفی شد. این روش، افتراق جمعیت را مستقیماً از داده‌های مولکولی، برآورد و فرضیات مربوط به این افتراق را آزمون می‌نماید. در تجزیه AMOVA، مجموع مربعات فواصل ژنتیکی به دو جزء بین‌گروهی و درون‌گروهی تقسیم می‌شود (جدول ۸-۱). آماره آزمون در این تجزیه به نام Φ_{ST} معادل آماره F رایت (Wright, 1965) می‌باشد. این آماره می‌تواند مقداری بین صفر و یک اختیار کند، که مقادیر بزرگتر، نشان‌دهنده‌ی افتراق بیشترِ جمعیتِ اصلی درجهت تشکیل زیرجمعیت‌ها است (برای اطلاعات بیشتر به منابع ذیل مراجعه نمایید؛ Excoffier *et al.*, 1992؛ Michalakis and Excoffier, 1996؛ Zhang and Ge, 2001؛ Mengoni and Bazzicalupo, 2002؛ Meirmans, 2006؛ Meirmans, 2012).

جدول ۸-۱-طرح کلی تجزیه سلسله مراتبی واریاس مولکولی (AMOVA)

منابع تغییر	درجه آزادی	میانگین مربعات	امید ریاضی میانگین مربعات
بین مکان‌ها	G-1	MSD/(AG)	$\sigma^2_c+n'\sigma^2_b+n''\sigma^2_a$
بین جمعیت‌ها در داخل مکان‌ها	$\sum_{g=1}^{G} I_g - G$	MSD/(AP/WG)	$\sigma^2_c+n\sigma^2_b$
بین افراد در داخل جمعیت‌ها	$N - \sum_{g=1}^{G} I_g$	MSD/(WP)	σ^2_c
کل	N-1		

به منظور شرح علایم، اختصارات و فرمول‌ها در جدول فوق، به منبع Excoffier *et al.*, 1992 مراجعه نمایید.

تجزیه AMOVA در R با تابع amova در پکیج‌هایpegas و ade4، با تابع adonis، در پکیج vegan و همچنین با تابع poppr.amova در پکیج poppr، قابل انجام است. در ذیل به شرح نحوه انجام تجزیه AMOVA توسط توابع مذکور می‌پردازیم.

تجزیه AMOVA توسط پکیج pegas

در تجزیه AMOVA باید از قبل یک نوع گروه‌بندی در مجموعه داده‌ها وجود داشته باشد و تجزیه AMOVA بر این اساس، واریانس کل دادها را در بین گروه‌ها و درون گروه‌ها تقسیم می‌نماید. در اینجا به

[27] Analysis of Molecular Variance

عنوان مثال، گروه‌بندی حاصل از تجزیه کلاستر K means که بر روی مجموعه داده nancycats انجام شد، مد نظر قرار می‌گیرد. (به منظور عدم نیاز به مراجعه به بخش تجزیه کلاستر K means، می‌توانید دستورات داخل کادر زیر را مجدداً اجرا نمایید. ولی اگر دستورات بخش تجزیه کلاستر K means را قبل از این بخش، اجرا کرده و از محیط R خارج نشده باشید، می‌توانید دستوراتی که بعد از کادر آمده است را اجرا نمایید).

```
> library(adegenet)
> data(dapcIllus)
> x <- dapcIllus$a
> grp <- find.clusters(x, max.n.clust=40)
        Choose the number PCs to retain (>=1): 150
        Choose the number of clusters (>=2: 6
```

برای انجام دستور amova از پکیج pegas نیاز به محاسبه و ایجاد دو شیء از قبل است. یکی از این اشیاء قالب داده‌ای است که حاوی اسامی جمعیت‌ها یا کلاسترهای انتسابی افراد است. این داده‌ها در جزء grp از شیء خروجی تابع find.clusters موجود است:

```
> head(grp$grp,20)
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 1  1  1  6  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
Levels: 1 2 3 4 5 6
```

(توجه داشته باشید از آنجا که در تجزیه کلاستر به روش K means گروه‌ها بطور تصادفی شماره‌گذاری می‌شوند، ممکن است نتایج حاصل شده برای شما با اعداد فوق متفاوت باشد ولی نتایج کلی تجزیه یکسان یکسان خواهد بود)

بنابراین در گام اول نسبت به تشکیل قالب داده از این شیء اقدام می‌نماییم:

```
> a<-data.frame(Kmeans=grp$grp)
> head(a)
  Kmeans
1      1
```

2	1
3	1
4	6
5	1
6	1

دومین ورودیِ مورد نیاز برای استفاده در تابع amova، شیءِ فاصله ژنتیکی بین افراد است که از توابع مختلفی که بدین منظور ایجاد شده‌اند، از جمله تابع dist در پکیج stats، قابل محاسبه می‌باشد:

```
> my.dist<-dist(dapcIllus$a)
> b<-as.matrix(my.dist)
> dim(b)
[1] 600 600
```

اکنون می‌توان تجزیه AMOVA را انجام داد. از آنجا که اسم تابع amova در هر دو پکیج pegas و ade4 مشابه است. در چنین حالاتی برای اینکه مشخص نماییم کدامیک از پکیج‌ها برای اجرای تابع، مد نظر می‌باشد، از علامت:: استفاده می‌کنیم. بنابراین دستور pegas::amova به معنای اینست که تابع amova از پکیج pegas، باید اجرا شود:

```
> library(pegas)
> set.seed(20170615)
> dapcIllus$a_amova<-pegas::amova(my.dist ~Kmeans, data = a, nperm =10)
> dapcIllus$a_amova

        Analysis of Molecular Variance

Call: pegas::amova(formula = my.dist ~ Kmeans, data = a, nperm = 10)

            SSD          MSD  df
Kmeans  5691.349   1138.26974   5
Error  13576.701     22.85640 594
```

```
Total       19268.050   32.16703      599

Variance components:
            sigma2    P.value
Kmeans 11.156       0
Error       22.856

Variance coefficients:
      a
99.98533
```

نتایج فوق نشان‌دهنده‌ی وجود تفاوت معنی‌دار بین گروه‌های از قبل تشکیل شده (براساس تجزیه کلاستر K means) است. از آنجا که این آزمون در تابع amova براساس نمونه‌برداری تصادفی انجام می‌شود، به منظور اخذ نتایج یکسان در صورت تکرار اجرای تجزیه، از دستور set.seed قبل از تابع amova استفاده شد. در این رابطه باید توجه داشت عددی که در تابع set.seed درج می‌شود اختیاری است، ولی برای تکرار نتایج، لازم است همان عدد را وارد کنیم وگرنه مقادیر عددیِ نتایج، یکسان نخواهد بود. بنابراین درصورتی که نیازمند به دسترسی به همین نتایج درصورت اجرای مجدد تابع amova باشیم، لازم است عدد مذکور را یادداشت نماییم و هنگام اجرای مجدد تجزیه، در تابع set.seed وارد کنیم.

نکته دیگر که باید توجه داشت اینست که آرگومان nperm در تابع amova، تعداد جایگشت‌ها را نشان می‌دهد که در اینجا برای محاسبه و مشاهده سریع نتایج، عدد آن کوچک (۱۰) انتخاب شد، ولی برای انجام تحقیق علمی لازم است عدد بزرگتری مانند ۹۹ یا غیره انتخاب شود که البته بدیهی است که با بزرگ شدن این عدد، مدت لازم برای انجام محاسبات نیز بیشتر خواهد شد.

تجزیه AMOVA توسط پکیج vegan

از اشیاء تشکیل شده قبل از اجرای تابع amova که در فوق شرح داده شد، می‌توان به عنوان ورودی تابع adonis نیز استفاده نمود و تجزیه را با استفاده از این تابع انجام داد:

```
> library(vegan)
> set.seed(20170615)
```

```
> dapcIllus$a_amova2<-adonis(my.dist ~Kmeans, data = a, permutations =99)
> dapcIllus$a_amova2

Call:
adonis(formula = my.dist ~ Kmeans, data = a, permutations = 99)

Permutation: free
Number of permutations: 99

Terms added sequentially (first to last)
```

	Df	SumsOfSqs	MeanSqs	F.Model	R2	Pr(>F)
Kmeans	5	5691.3	1138.27	49.801	0.29538	0.01**
Residuals	594	13576.7	22.86		0.70462	
Total	599	19268.1			1.00000	

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

تجزیه AMOVA توسط پکیج poppr

نحوه انجام تجزیه AMOVA توسط تابع poppr.amova در پکیج poppr را به عنوان مثال بر روی مجموعه داده microbov توضیح می‌دهیم. قبل از آن لازم است با تابع strata، الگوی گروه‌بندی موجود در بخش other از این مجموعه داده را به افراد منتسب نماییم:

```
> library(poppr)
> data(microbov)
> strata(microbov) <- data.frame(other(microbov))
> head(strata(microbov))
```

	coun	breed	spe
1	AF	Borgou	BI
2	AF	Borgou	BI
3	AF	Borgou	BI
4	AF	Borgou	BI
5	AF	Borgou	BI
6	AF	Borgou	BI

با در نظر گرفتن "کشور" به عنوان گروه اصلی و "نژاد" به عنوان گروه فرعی، تجزیه AMOVA توسط تابع poppr.amova بصورت زیر قابل انجام است:

```
> mic.amova <- poppr.amova(microbov, ~coun/breed)

        Found 12650 missing values.

        2 loci contained missing values greater than 5%

        Removing 2 loci: ILSTS6, ETH185

        Distance matrix is non-euclidean.

        Utilizing quasieuclid correction method. See ?quasieuclid for details.
```

همانطور که مشاهده می‌شود تابع poppr.amova ابتدا گزارشی از مقادیر گمشده ارائه می‌دهد و برای انجام تجزیه، لوکوس‌های دارای مقدار گمشده را از تجزیه حذف می‌نماید. نتایج تجزیه AMOVA در شیء خروجی تابع poppr.amova قابل مشاهده می‌باشد:

```
> mic.amova

        $call

        ade4::amova(samples = xtab, distances = xdist, structures = xstruct)

        $results
```

	Df	Sum Sq	Mean Sq
Between coun	1	732.8379	732.8379
Between breed Within coun	13	1089.227	83.78667
Between samples Within breed	689	6513.337	9.453319
Within samples	704	6010.457	8.537582
Total	1407	14345.86	10.19606

```
        $componentsofcovariance
```

	Sigma	%
Variations Between coun	1.041929	9.618112
Variations Between breed Within coun	0.795609	7.344314
Variations Between samples Within breed	0.457869	4.226616
Variations Within samples	8.537582	78.81096
Total variations	10.83299	100

```
$statphi
```

	Phi
Phi-samples-total	0.21189
Phi-samples-breed	0.0509
Phi-breed-coun	0.081259
Phi-coun-total	0.096181

شبیه سازی هیبریداسیون

یکی از توانمندی‌های برنامه R در حوزه ژنتیک جمعیت، امکان شبیه سازی و پیش‌بینی نتایج هیبریداسیون بین جمعیت‌ها می‌باشد که در پکیج adgenet تعبیه شده است. تابع hybridize می‌تواند عمل هیبریداسیون بین دو مجموعه از ژنوتیپ‌هایی که در قالب شیء genind ذخیره شده‌اند را انجام دهد. برای این کار، تابع hybridize فراوانی‌های آللی در هر جمعیت را معیار قرار می‌دهد و برمبنای یک توزیع چند جمله‌ای، از گامت‌ها نمونه‌برداری می‌کند. به عنوان مثال برای شبیه‌سازی هیبریداسیون بین جمعیت‌های Salers و Zebu، از مجموعه داده‌های microbov، ابتدا دو جمعیت را در قالب دو شیء genind مجزا می‌نماییم:

```
> library(adegenet)
> data(microbov)
> temp <- seppop(microbov)
> names(temp)
 [1] "Borgou"         "Zebu"           "Lagunaire"      "NDama"
 [5] "Somba"          "Aubrac"         "Bazadais"       "BlondeAquitaine"
 [9] "BretPieNoire"   "Charolais"      "Gascon"         "Limousin"
[13] "MaineAnjou"     "Montbeliard"    "Salers"
> salers<- temp$Salers
> zebu<- temp$Zebu
```

اکنون می‌توان عمل شبیه سازی هیبریداسیون بین جمعیت‌ها را با استفاده از تابع hybridize انجام داد:

```
> F1 <- hybridize(salers, zebu, n=40, pop="zebler")
```

آرگومان n در تابع hybridize، تعداد افراد هیبرید در جمعیت جدید و آرگومان pop، اسم جمعیت جدید را مشخص می‌نماید. جمعیت جدید (خروجی تابع hybridize) خود یک شیء genind متشکل از افراد هیبرید به تعداد مشخص شده در آرگومان n، خواهد بود:

```
> F1
/// GENIND OBJECT /////////
 // 40 individuals; 30 loci; 282 alleles; size: 75.6 Kb
 // Basic content
   @tab:  40 x 282 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 5-17)
   @loc.fac: locus factor for the 282 columns of @tab
```

@all.names: list of allele names for each locus

@ploidy: ploidy of each individual (range: 2-2)

@type: codom

@call: hybridize(x1 = salers, x2 = zebu, n = 40, pop = "zebler")

// Optional content

@pop: population of each individual (group size range: 40-40)

```
> tab(F1)[1:5,1:5]
```

	BM1818.248	BM1818.254	BM1818.256	BM1818.258	BM1818.260
h01	0	0	0	0	0
h02	0	0	0	0	0
h03	0	0	0	0	0
h04	0	0	0	0	0
h05	0	0	0	0	0

همچنین می‌توان با انجام هیبریداسیون بین جمعیت جدید و والدین اولیه، نسل‌های تلاقی برگشتی و یا با انجام هیبریداسیون بین خود جمعیت‌های هیبرید، نسل‌های خودگشنی را شبیه‌سازی نمود:

```
> F2 <- hybridize(F1, F1, n=100)
> F3 <- hybridize(F2, F2, n=100)
> Bc1 <- hybridize(salers, F1, n=100)
> Bc2 <- hybridize(zebu, F1, n=100)
> Bc1F1 <- hybridize(Bc1, Bc1, n=100)
> Bc1F2 <- hybridize(Bc1F1, Bc1F1, n=100)
```

فصل نهم-شناسایی الگوهای ژنتیکی- جغرافیایی

انواع روش‌های بررسی تغییرات ژنتیکی وابسته به مکان

همانطور که در بخش‌های پیشین توضیح داده شد برای نمایش تنوع ژنتیکی، بطور معمول از نمودارهای مبتنی بر تجزیه‌های چندمتغیره مانند تجزیه به مؤلفه‌های اصلی، تجزیه به مختصات اصلی و غیره، استفاده می‌شود. این روش‌ها به ویژه برای گروه‌بندی نمونه‌ها و شناسایی اعضاء خارج از گروه، مناسب می‌باشند. از آنجا که در این روش‌های تجزیه، داده‌های جغرافیایی مربوط به مکان مشاهده یا نمونه‌برداری، منظور نمی‌گردد، دسته‌بندی‌های صورت گرفته روی ژن‌ها، جمعیت‌ها، گونه‌ها و غیره، در فضایی مجازی به صورت نمودار نمایش داده می‌شود. اما چنانچه بخواهیم الگوهای ژنتیکی را با درنظر گرفتن پراکنش جغرافیایی نمونه‌ها شناسایی کنیم، لازم است روش‌های دیگری را بکار گیریم. جغرافیای ژنتیکی[۲۸] یا جغرافیای ژنی[۲۹] به عنوان یک زمینه علمیِ بین رشته‌ای، برای نخستین بار توسط یک متخصص ژنتیک روسی تبار به نام سربروسکی در سال ۱۹۲۸ معرفی شد. از آن زمان، به منظور مطالعه تنوع ژنتیکی در زمینه‌ی فواصل جغرافیایی، چندین روش تجزیه توسعه یافته است. مالکوت (Malecot, 1948) مدل انزوای ناشی از فاصله[۳۰] (IBD) را برای بررسی ارتباط بین ماتریس‌های فواصل جغرافیای و فواصل ژنتیکی ارئه کرد. این روش مبتنی بر تجزیه رگرسیون و همبستگی می‌باشد. منظور از IBD، فرایندی است که در آن به دلیل محدودیت جغرافیایی برای جریان ژنی، پدیده رانده شدن ژنتیکیِ موضعی رخ می‌دهد و ساختار ژنتیکی ایجاد می‌شود. توجه به این پدیده در زمانی که ساختارهای ژنتیکی و روندهای تکاملی، وابستگی مکانی نشان دهند، از

[28] Genogeography
[29] Gene geography
[30] Isolation by distance

اهمیت برخوردار است. هنگامی که تبادل ژنی در زیرواحدهای جمعیت با نرخی وابسته به مسافت رخ دهد، پدیده IBD مشاهده خواهد شد. مبانی تجزیه نظری IBD، توسط رایت (Malecot, 1946) و مالکوت (Malecot, 1948) ارائه شد. مالکوت نشان داد که چگونه میزان خویشاوندیِ افراد، تحت تأثیر فاصله بین آنها قرار می‌گیرد و از آن زمان مؤلفان زیادی از این روش برای توصیف چگونگی توسعه ساختارهای ژنتیکی در حالت‌های مختلف IBD، استفاده نموده‌اند. خودهمبستگی مکانی[۳۱]، روشی دیگر برای بررسی ارتباط بین فواصل جغرافیای و فواصل ژنتیکی است. اگرچه از این روش بطور گسترده‌ای در شناسایی IBD استفاده شده است، اما اشکال آن اینست که هیچ‌گونه اطلاعاتی از الگوی ویژه‌ی تنوع در یک فضای دو بعدی ارائه نمی‌کند. این روش توسط افراد مختلفی مورد انتقاد قرار گرفته است از جمله اینکه آلل‌ها و مکان‌های ژنی در معرض نیروهای تکاملی متفاوتی، بویژه نرخ جهش و فشار گزینشی متفاوت، می‌باشند. لذا میانگین خودهمبستگی بر روی مکان‌های ژنی یا آلل‌ها، بی‌معنی می‌باشد. روش دیگر، مبتنی بر تجزیه به مؤلفه‌های اصلی است. در این روش سه مؤلفه اصلیِ اول بطور جداگانه، به عنوان محور z نقشه‌ها در نظر گرفته می‌شوند و محورهای x و y نیز عبارت از مختصات جعرافیایی می‌باشند. در این روش فقط چشم‌انداز ژنتیکی در فضای جغرافیایی به تصویر کشیده می‌شود، اما الگوی تغییرات فراوانی‌های ژنتیکی، مورد تجزیه آماری قرار نمی‌گیرد. روشی دیگر که توسط ومبل (Womble, 1951) معرفی و توسط سایر محققان توسعه یافت، مبتنی بر تعیین مرزها یا موانع است که در آن بر جزئیات جغرافیایی در نواحی که تغییرات ژنتیکی قابل توجهی رخ می‌دهد، تمرکز می‌شود. مانمونیر (Monmonier, 1973) الگوریتمی ارائه داد که قادر است تغییرات شدید را در یک ماتریس فاصله، شناسایی کرده و براساس آنها، نقشه جغرافیایی ترسیم نماید. این فواصل می‌توانند ژنتیکی، مورفولوژیکی یا هر چیز دیگری باشند. الگوریتم مانمونیر در زمینه مطالعات ژنتیکی توسط محققان دیگری مورد استفاده قرار گرفت و تجزیه بوتسرپ برای آزمون معنی‌دار بودن مرزهای ژنتیکی شناسایی شده، توسعه یافت (Womble, 1951؛ Kimura, and Weiss, 1964؛ Imaizumi, *et al.*,1970؛ Gabriel, 1971؛ Monmonier, 1973؛ Sokal, and Oden, 1978؛ Levin, 1981؛ Sokal, and Wartenberg, 1983؛ Barbujani, 1987؛ Cockerham and Weir, 1987؛ Barbujani *et al.*,1989؛ Sokal *et al.*, 1989؛ Barbujani *et al.*, 1990؛ Heywood, 1991؛ Slatkin, and Arter, 1991؛ Oden *et al.*,1993؛ Sokal *et al.*,1993؛ Epperson, 1993؛ Slatkin, 1993؛ Bocquet-Appel and Bacro, 1994؛ Jacquez, 1995؛ Epperson, 1995؛ Hamrick *et al.*,1995؛ Barbujani *et al.*,1996؛ Epperson and Li, 1996؛ Rousset, 1997؛ Smouse, and Peakall, 1999؛ Sokal *et al.*,1999؛ Manni and Barrai, 2001؛ Nielsen, 2001؛ Manni *et al.*, 2002؛ Dupanloup *et al.*, 2002؛ Manel *et al.*, 2003؛ و Manni *et al.*, 2004).

[31] Spatial autocorrelation

نرم‌افزار R جامع‌ترین مجموعه از روش‌های تجزیه داده‌های مکانی را در بین نرم‌افزارهای آماری ارائه می‌دهد. در اینجا دو روشی که بطور عمومی در ژنتیک جمعیت مورد استفاده قرار می‌گیرد که عبارتند از آزمون مانتل (Mantel, 1967) برای بررسی وجود پدیده انزوای ناشی از فاصله (IBD) و الگوریتم مانمونیر (Monmonier, 1973) برای تعیین مرزهای ژنتیکی، توضیح داده می‌شود.

آزمون مانتل

همبستگی بین فواصل ژنتیکی و فواصل جغرافیایی می‌تواند تحت سناریوهای زیستی متفاوتی رخ دهد. پدیده انزوای ناشی از فاصله کلاسیک (IBD) منجر به شیب پیوسته افتراق ژنتیکی می‌شود و چنین همبستگی را ایجاد می‌کند. اما جمعیت‌های متفرق و دور از یکدیگر نیز، چنین الگوهایی را ایجاد می‌کنند. این فرآیندها تا حدی متفاوت از یکدیگر می‌باشند و بایستی از یکدیگر تمیز داده شوند.

انزوای ناشی از فاصله کلاسیک (IBD) را می‌توان با استفاده از آزمون مانتل[۳۲] برای همبستگی بین ماتریس فواصل ژنتیکی و ماتریس فواصل جغرافیایی ارزیابی نمود. این آزمون هم برای افراد و هم برای جمعیت‌ها، قابل انجام می‌باشد.

مثال: می‌خواهیم در مجموعه داده nancycats، همبستگی بین فواصل ژنتیکی جمعیت‌ها (براساس روش Edwards) و فواصل جغرافیایی اقلیدسی را محاسبه می‌کنیم. بطور کلی مختصات مکانی جمعیت‌ها در یک شیء genind مانند obj، در جزء xy از بخش other قرار دارد و با دستور other(obj)$xy یا obj$other$xy، قابل فراخوانی و مشاهده می‌باشد:

```
> library(adegenet)
>  data(nancycats)
> nancycats$other$xy
```

	x	y
P01	263.3498	171.10939
P02	183.5028	122.4079
P03	391.105	254.70148
P04	458.6121	41.72336
P05	182.7769	219.08398
P06	335.2121	344.83557
P07	359.1662	375.36486

[32] Mantel

P08	271.3345	67.89132
P09	256.8169	150.02964
P10	270.6086	17.00917
P11	493.4544	237.25618
P12	305.451	85.33663
P13	462.9674	86.7904
P14	429.5768	291.04587
P15	531.2003	115.13903
P16	407.8003	99.87438
P17	345.3745	251.79393

محاسبه فواصل جغرافیایی بین جمعیت‌ها به روش اقلیدسی با تابع dist قابل انجام است:

```
> Dgeo <- dist(nancycats$other$xy)
> class(Dgeo)
[1] "dist"
> as.matrix(Dgeo)[1:5,1:5]
```

	P01	P02	P03	P04	P05
P01	0.00000	93.52743	152.67300	234.23940	93.77395
P02	93.52743	0.00000	246.17120	286.69690	96.67880
P03	152.67297	246.17124	0.00000	223.42090	211.35092
P04	234.23938	286.69688	223.42090	0.00000	327.93569
P05	93.77395	96.67880	211.35090	327.93570	0.00000

برای محاسبه فواصل ژنتیکی بین جمعیت‌ها لازم است ابتدا شیء genind را genpop به تبدیل کنیم:

```
> p.cats <- genind2genpop(nancycats)
> Dgen <- dist.genpop(p.cats, method=2)
> class(Dgen)
[1] "dist"
> as.matrix(Dgen)[1:5,1:5]
```

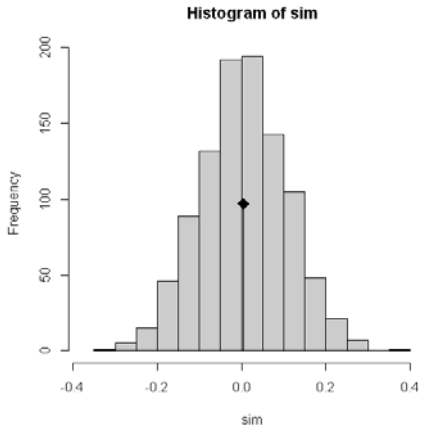
	P01	P02	P03	P04	P05
P01	0.0000000	0.5957681	0.5277615	0.5245342	0.4814277
P02	0.5957681	0.0000000	0.5831500	0.5499330	0.5635837
P03	0.5277615	0.5831500	0.0000000	0.3746269	0.4153192
P04	0.5245342	0.5499330	0.3746269	0.0000000	0.4277317
P05	0.4814277	0.5635837	0.4153192	0.4277317	0.0000000

در دستور فوق، آرگومان method=2 به روش Edwards اشاره دارد. همبستگی بین دو ماتریس فاصله به روش مونت کارلو توسط تابع mantel.randtest قابل آزمون است:

```
> man.tst <- mantel.randtest(Dgen,Dgeo)
> man.tst
        Monte-Carlo test
        Call: mantel.randtest(m1 = Dgen, m2 = Dgeo)
        Observation: 0.00492068
        Based on 999 replicates
        Simulated p-value: 0.502
        Alternative hypothesis: greater
           Std.Obs  Expectation     Variance
        -0.004745247  0.005403482  0.010351878
```

مقدار بزرگ p-value نشان‌دهنده‌ی عدم وجود همبستگی معنی‌دار بین فواصل ژنتیکی و فواصل جغرافیایی می‌باشد. نتایج این آزمون را می‌توان با استفاده از نمودار نیز نشان داد (شکل ۹-۱):

```
> plot(man.tst)
```

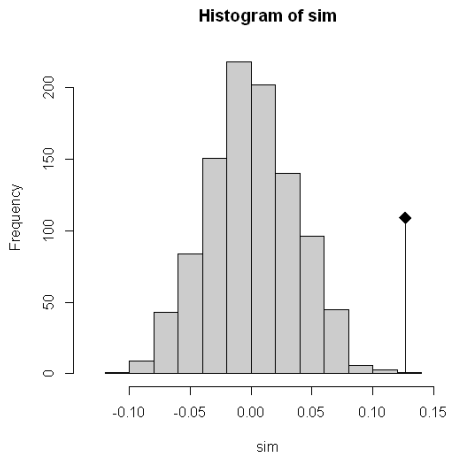


شکل ۹-۱- توزیع مرجع برای آزمون همبستگی مانتل بین ماتریس‌های فاصله ژنتیکی و جغرافیایی به روش مونت کارلو در مجموعه داده nancycats

مقدار همبستگی بین دو ماتریس فاصله در نمودار، با نقطه نشان داده شده است و هیستوگرام، حاصل از مقادیری است که بطور تصادفی، یعنی بدون در نظر گرفتن ساختارهای مکانی ایجاد شده‌اند. اگر ساختارهای مکانی معنی‌دار بود، مقدار همبستگی بین دو ماتریس فاصله باید خارج از توزیع مرجع قرار می‌گرفت. این نتایج نشان می‌دهد که IBD، در مجموعه داده nancycats معنی‌دار نمی‌باشد.

در مثالی دیگر همبستگی بین فواصل ژنتیکی جمعیت‌ها و فواصل جغرافیایی در داده‌های dat2B از مجموعه spcaIllus ، که تحت مدل IBD شبیه‌سازی شده است را بررسی می‌نماییم (شکل ۹-۲):

```
> data(spcaIllus)
> x <- spcaIllus$dat2B
> Dgen <- dist(x$tab)
> Dgeo <- dist(other(x)$xy)
> man.test<- mantel.randtest(Dgen,Dgeo)
> man.test
        Monte-Carlo test
        Call: mantel.randtest(m1 = Dgen, m2 = Dgeo)
        Observation: 0.1267341
        Based on 999 replicates
        Simulated p-value: 0.002
        Alternative hypothesis: greater

           Std.Obs  Expectation     Variance
        3.4539482127 0.0002584846 0.0013408556
> plot(man.test)
```



شکل ۹-۲ - توزیع مرجع برای آزمون همبستگی مانتل بین ماتریس‌های فاصله ژنتیکی و جغرافیایی به روش مونت کارلو در داده‌های dat2B از مجموعه spcaIllus

نتایج آزمون و نمودار، وجود همبستگی معنی‌دار بین فواصل ژنتیکی و فواصل جغرافیایی جمعیت‌ها و در نتیجه وجود IBD را نشان می‌دهد.

مرزهای ژنتیکی

تعیین مرز جغرافیایی بین جمعیت‌های ژنتیکی که مختصات جغرافیایی آنها مشخص شده است، می‌تواند به عنوان معیاری از تأثیر فاصله جغرافیایی بر خصوصیات ژنتیکی جمعیت، مدنظر قرار گیرد. برای تعیین این حدود، می‌توان از الگوریتمی که مانمونیر (Monmonier, 1973) ارائه داد، استفاده نمود. این روش در ابتدا برای تجزیه‌های جغرافیایی مورد استفاده قرار می‌گرفت. مانی و همکاران (Manni *et al.*, 2004) پیشنهاد دادند که از این الگوریتم، می‌توان برای شناسایی مرزهای بین ژنوتیپ‌ها یا جمعیت‌ها، براساس مختصات جغرافیایی آنها استفاده نمود. هدف از الگوریتم Monmonier، شناسایی مسیری است که از بزرگترین فواصل ژنتیکی بین نقاط همسایه عبور می‌کند. در این الگوریتم بیشترین فاصله ژنتیکی بین نقاط همجوار به عنوان نقطه آغاز مسیر مرز در نظر گرفته می‌شود، سپس نقاط همجوار با این نقطه، مدنظر قرار گرفته و مسیر مرز از بین نقاطی با بیشترین فاصله ژنتیکی امتداد می‌یابد. این فرایند تا رسیدن به مقدار آستانه معینی ادامه می‌یابد. منظور از تعیین آستانه، تفکیک فواصل ژنتیکی بین نقاط واقع در درون یک مرز از فواصل ژنتیکی بین نقاط واقع در دو طرف مرز می‌باشد. برای مشخص شدن مقدار عددی آستانه، فواصل ژنتیکی بین نقاط

همجوار به ترتیبِ اندازه در نموداری نشان داده می‌شوند و جایی که افت ناگهانی در فواصل ژنتیکی مشاهده شود، به عنوان مقدار آستانه در نظر گرفته می‌شود.

مثال: مجموعه داده sim2pop متشکل از دو جمعیت شبیه‌سازی شده با اندازه‌های ۳۰ و ۱۰۰ فرد است. ابتدا تصویری از پراکنش جغرافیایی جمعیت‌ها ارائه می‌دهیم:

```
> data(sim2pop)
> summary(sim2pop$pop)
    P01 P02
    100  30
> mypop <- sim2pop$pop
> mypop
    [1] P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01
   [19] P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01 P01
   ..........................................
  [109] P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02 P02
  [127] P02 P02 P02 P02
  Levels: P01 P02
> levels(mypop) <- c(3,5)
> mypop
    [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
   [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
   [75] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
  [112] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
```

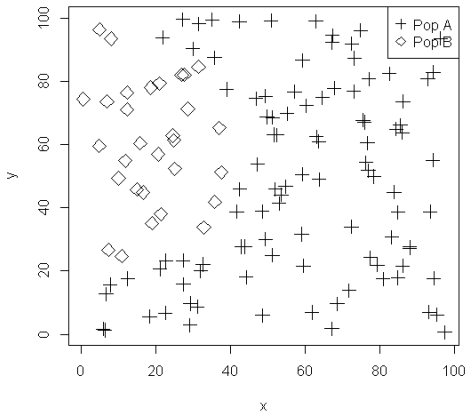
همانطور که مشاهده می‌شود سطوح تعریف شده برای جمعیت‌ها، در ابتدا P01 و P02 بود . با استفاده از تابع levels، سطوح جمعیت‌ها را در متغیر جدید (mypop) بصورت 3 و 5 مجدداً نامگذاری کردیم تا بعداً بتوانیم از آنها به عنوان کاراکتر در نمودار (شکل ۹-۳) استفاده کنیم (در آرگومان pch، 3 مربوط به نماد + و 5 مربوط به نماد ◊ است)

```
> class(mypop)
```

```
[1] "factor"
> mypopn <- as.numeric(as.character(mypop))
> mypopn
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [38] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [75] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
[112] 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
```

نکته: تبدیل شیء mypop از کلاس فاکتور به کاراکتر (توسط تابع as.character)، قبل از تبدیل نهایی به کلاس عددی (توسط تابع as.numeric) ضروری است، زیرا در غیراینصورت (یعنی تبدیل کلاس فاکتور به عددی، بطور مستقیم) اعداد ۳ و ۵ در شیء mypop در نظر گرفته نمی‌شوند، و بجای آنها اعداد ۱ و ۲ که در متغیر mypopn جایگزین شوند، این موضوع را می‌توانید با اجرای مجدد دستورات و حذف مرحله مربوطه امتحان نمایید یعنی بجای دستور (as.numeric(as.character(mypop))، دستور as.numeric(mypop) را اجرا کنید).

```
> plot(sim2pop$other$xy,pch=mypopn,cex=1.5)
> legend("topright",leg=c("Pop A", "Pop B"),pch=c(3,5))
```



شکل ۹-۳ – پراکنش جغرافیایی دو جمعیت شبیه‌سازی A و B از مجموعه داده sim2pop

اکنون از تابع monmonier به شرح زیر برای ترسیم مرز ژنتیکی استفاده می‌نماییم:

monmonier (xy, dist, cn)

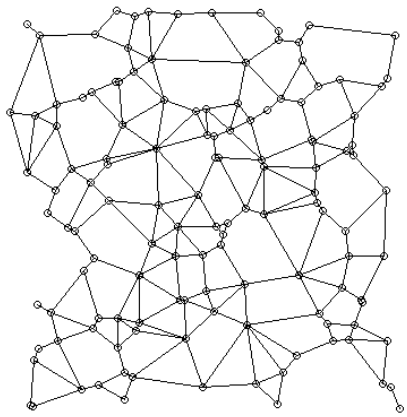
همانطور که مشاهده می‌شود، تابع monmonier دارای سه آرگومان اصلی xy، dist و cn است. xy عبارت از ماتریس مختصات جغرافیایی است که از قبل، در شیء genind ذخیره شده است. dist عبارت از ماتریس فواصل ژنتیکی جفتی است و cn شبکه ارتباطات می‌باشد. بنابراین برای اجرای تابع monmonier نیاز است که ابتدا فواصل ژنتیکی و شبکه ارتباطات بدست آید. برای محاسبه فواصل ژنتیکی می‌توان از تابع dist استفاده کرد:

```
> D <- dist(sim2pop$tab)
```

برای به دست آوردن شبکه ارتباطات (شکل ۹-۴)، می‌توان از تابع chooseCN استفاده نمود:

```
> gab <- chooseCN(sim2pop$other$xy,ask=FALSE,type=2)
```

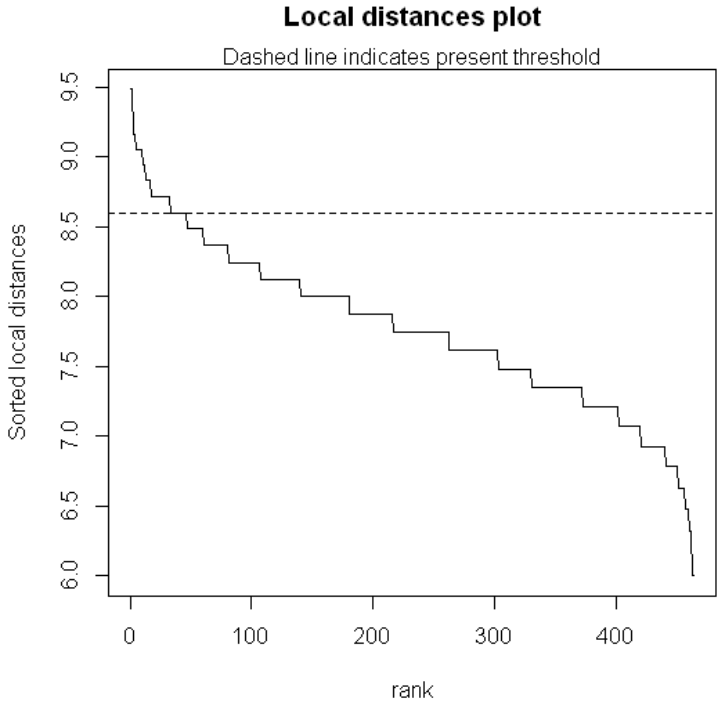
در دستور فوق با انتخاب آرگومان type=2 شبکه از نوع Gabriel تعیین شده و با گزینه ask=FALSE پرسش از کاربر، غیرفعال شده است.



شکل ۹-۴ – شبکه ارتباطات بین نقاط همجوار از نوع Gabriel برمبنای اطلاعات جغرافیایی مجموعه داده sim2pop

اکنون می‌توان تابع monmonier را اجرا نمود:

```
> mon1 <- monmonier(sim2pop$other$xy,D,gab)
```



شکل ۹-۵ - نمایش فواصل ژنتیکی به ترتیب نزولی به منظور تعیین مقدار آستانه در الگوریتم monmonier بر روی مجموعه داده sim2pop

در نمودار شکل ۹-۵ فواصل ژنتیکی بصورت نزولی مرتب شده است. درصورت وجود مرز معنی‌دار، بایستی این مقادیر پس از آن مرز، افت ناگهانی نشان دهد که دراینصورت به عنوان "مقدار آستانه " برای الگوریتم، در نظر گرفته می‌شود تا پیشبرد مرز در ورای آن را متوقف نماید. اما در نمودار فوق، افت ناگهانی در بین مقادیر مربوطه چندان مشهود نیست و لذا در وهله اول بنظر می‌رسد چنین مرزی وجود ندارد.

اکنون سؤال این است که چرا الگوریتم، در شناسایی مرز، ناموفق عمل کرده است؟ دلیل این امر یا اینست که درواقع افتراق ژنتیکی وجود ندارد و یا اینکه علائم تفکیک کننده جمعیت‌ها آنقدر ضعیف است که نمی‌تواند بر پارازیت تصادفی در فواصل ژنتیکی چیره شود. برای بررسی این موضوع، پارامتر F_{ST} را به عنوان معیار افتراق جمعیت‌ها، مدنظر قرار می‌دهیم:

```
> library(hierfstat)
> pairwise.fst(sim2pop)
          1
2 0.02960988
```

میزان $F_{ST} = 0.0296$ در ظاهر نشان‌دهنده‌ی افتراق ضعیفی می‌باشد. برای بررسی معنی‌دار بودن آماری این مقدار، می‌توان آزمون جایگشت انجام داد:

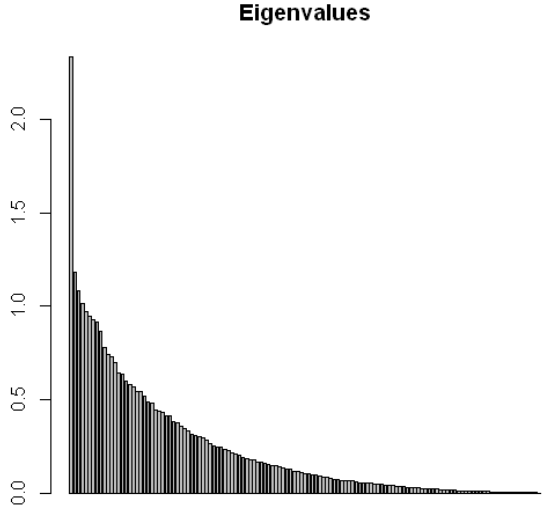
```
> replicate(10, pairwise.fst(sim2pop, pop=sample(pop(sim2pop))))
```

```
[1] 0.003901177 0.004230156 0.003707953 0.004923095 0.003837047 0.003498073
[7] 0.003149938 0.003832049 0.003166595 0.003040000
```

در دستور فوق به منظور صرفه جویی در زمان لازم برای انجام محاسبات، تعداد کمی جایگشت (۱۰ جایگشت) در نظر گرفته شد، ولی شما می‌توانید آن را با تعداد زیادتر (مثلاً ۲۰۰ جایگشت) انجام دهید. با اینحال، در همین نتایج نیز مشخص است که مقدار F_{ST} مشاهده شده (۰/۰۲۳)، به مراتب از ده مقدار تصادفی که با فرض عدم وجود ساختار در جمعیت حاصل شده‌اند، بزرگتر است. لذا این نتایج نشان‌دهنده‌ی وجود افتراق ژنتیکی بین جمعیت‌ها است، اگرچه F_{ST} مشاهده شده در ظاهر کوچک به نظر می‌رسد. لازم به یادآوری است که مقادیر کوچک F_{ST} حتی در صورت وجود میزان بالای افتراق بین جمعیت‌ها، در زمره ایراداتی است که به این کمیّت وارد می‌شود که در بخش آماره‌های افتراق ژنتیکی نیز پیرامون آن توضیح داده شد.

این مشاهدات حاکی از احتمال وجود پارازیتی است که مانع تشخیص مرز ژنتیکی معنی‌دار بین جمعیت‌ها شده است. بنابراین اگر بتوانیم این پارازیت را از داده‌ها حذف کنیم، شاید الگوریتم Monmonier قادر به تعیین مرز ژنتیکی معنی‌دار بین دو جمیعت باشد. این کار با تجزیه به مختصات اصلی قابل انجام است (شکل ۹-۶):

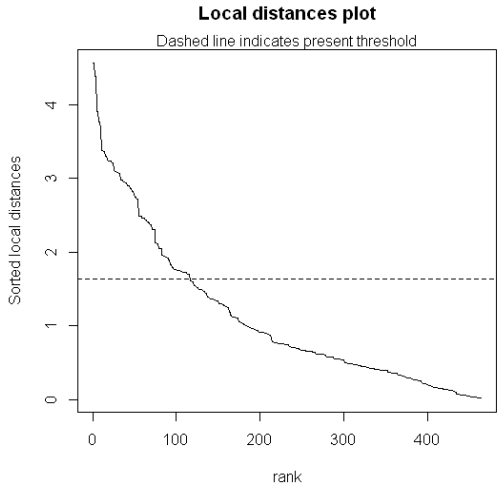
```
> pco1 <- dudi.pco(D,scannf=FALSE,nf=1)
> barplot(pco1$eig,main="Eigenvalues")
```



شکل ۹-۶ – نمودار ستونی مقادیر ویژه مربوط به تجزیه به مختصات اصلی مبتنی بر فواصل ژنتیکی به دست آمده از مجموعه داده sim2pop

در تجزیه فوق فقط مقدار ویژه اول را نگه داشتیم و با استفاده از مختصات اصلی مربوطه، فاصله ژنتیکی بین ژنوتیپ‌ها را مجددا تعریف می‌کنیم (شکل ۹-۷):

```
> D2 <- dist(pco1$li)
> mon2 <- monmonier(sim2pop$other$xy,D2,gab)
```



شکل ۹-۷ – نمایش فواصل ژنتیکی به ترتیب نزولی در الگوریتم monmonier بعد از انجام تجزیه به مختصات اصلی بر روی مجموعه داده sim2pop

مقدار آستانه تفاوت که در نمودار مشخص شده، بین اعداد ۱ و ۲، با کمی تمایل به طرف ۲ است، لذا در جواب پرسش از کاربر، مقدار ۱/۶ را وارد می‌نماییم:

```
Indicate the threshold ('d' for default): 1.6
```

خروجی تابع شیئی از نوع monmonier است:

```
> class(mon2)
[1] "monmonier"

> mon2

        ############################################################
        # List of paths of maximum differences between neighbours #
        #          Using a Monmonier based algorithm               #
        ############################################################

        $call:monmonier(xy = sim2pop$other$xy, dist = D2, cn = gab)

            # Object content #
        Class:  monmonier
        $nrun (number of successive runs):  1
        $run1: run of the algorithm
        $threshold (minimum difference between neighbours):  1.6
        $xy: spatial coordinates
        $cn: connection network
```

```
    # Runs content #
# Run 1
# First direction
Class:  list
$path:
           x       y
Point_1 14.98299 93.81162

$values:
 4.563555
# Second direction
Class:  list
$path:
           x       y
Point_1 14.98299 93.81162
Point_2 30.74508 87.57724
Point_3 33.66093 86.14115
...

$values:
 4.563555 3.23581 3.906439 ...
```

شیء monmonier، لیستی متشکل از اجزاء مختلف است که شامل تمام اطلاعات مربوط به مرزهای شناسایی شده است. از آنجا که چندین مرز می‌تواند بطور همزمان توسط آرگومان nrun مورد جستجو واقع شود، باید مشخص کنیم که می‌خواهیم اطلاعات مسیر را طبق کدام مرز (run) و در کدام جهت، بدست آوریم. به عنوان مثال:

```
> names(mon2)
        [1] "run1"      "nrun"      "threshold" "xy"        "cn"        "call"
> names(mon2$run1)
        [1] "dir1" "dir2"
> mon2$run1$dir1
        $path
                   x       y
        Point_1 14.98299 93.81162

        $values
        [1] 4.563555
```

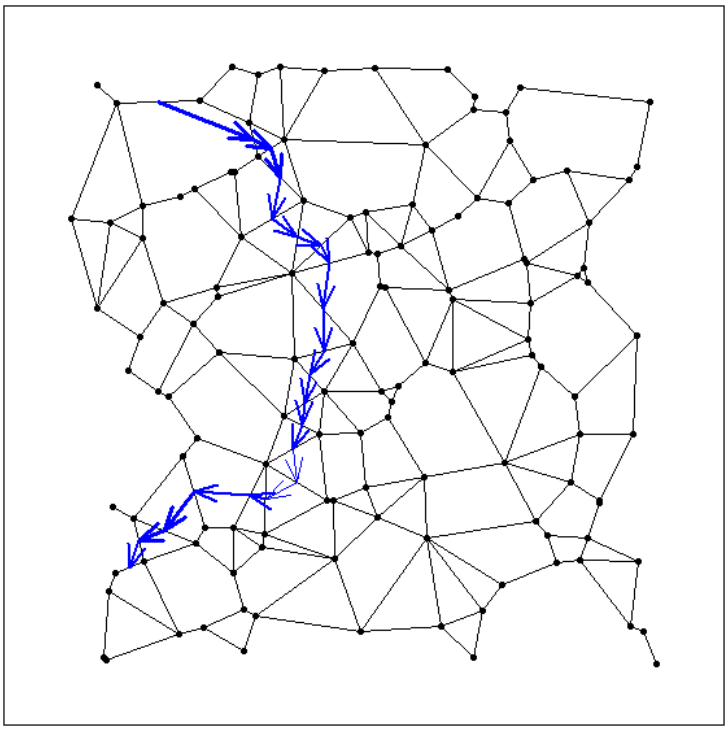
همچنین با استفاده از تابع coords.monmonier، می‌توان نقاطی را که خط مرزی از آنها عبور می‌کنند را شناسایی نمود:

```
> coords.monmonier(mon2)
        $run1
        $run1$dir1
                 x.hw     y.hw first second
        Point_1 14.98299 93.81162    11    125

        $run1$dir2
```

	x.hw	y.hw	first	second
Point_1	14.98299	93.81162	11	125
Point_2	30.74508	87.57724	44	128
Point_3	33.66093	86.14115	20	128
Point_4	35.28914	81.12578	68	128
Point_5	33.85756	74.45492	68	117
Point_6	38.07622	71.47532	68	122
Point_7	41.97494	70.02783	35	122
Point_8	43.45812	67.12026	69	122
Point_9	42.20206	59.59613	22	122
Point_10	42.48613	52.55145	22	124
Point_11	40.08702	48.61795	13	124
Point_12	39.20791	43.89978	13	127
Point_13	38.81236	40.34516	62	127
Point_14	37.32112	36.35265	62	130
Point_15	37.96426	30.82105	94	130
Point_16	32.79703	28.00517	16	130
Point_17	30.12832	28.60376	85	130
Point_18	20.92496	29.21211	63	119
Point_19	16.05811	22.726	61	126
Point_20	11.72524	21.15519	89	126
Point_21	10.18696	16.61536	74	89

با استفاده از تابع plot، می‌توان مرز را ترسیم نمود (شکل ۹-۸):

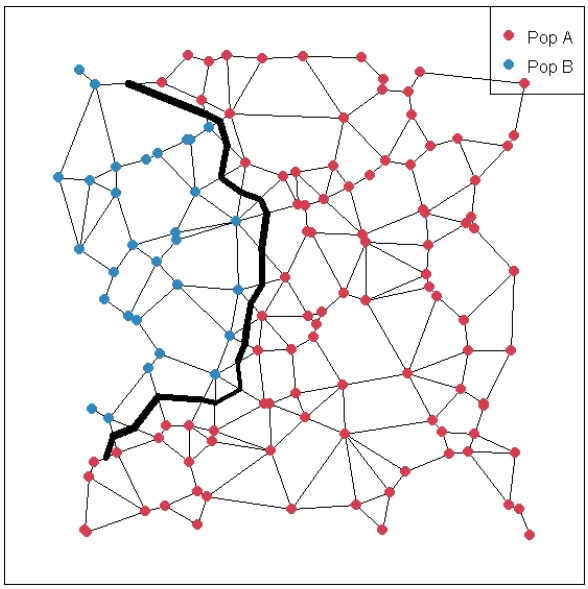
```
> plot(mon2)
```



شکل ۹-۸ – مرز ژنتیکی جدا کننده‌ی دو جمیعت شبیه سازی شده در مجموعه داده sim2pop

با اجرای دستورات زیر می‌توان مرز بدست آمده را با پراکنش جمعیت‌ها، بطور توأم در یک نمودار مورد مقایسه و تطابق قرار داد (شکل ۹-۹):

```
> plot(mon2,add.arrows=FALSE,bwd=10,col="black")
> points(sim2pop$other$xy, cex=2, pch=20, col=fac2col(pop(sim2pop), col.pal=spectral))
> legend("topright",leg=c("Pop A", "Pop B"), pch=c(20), col=spectral(2), pt.cex=2)
```

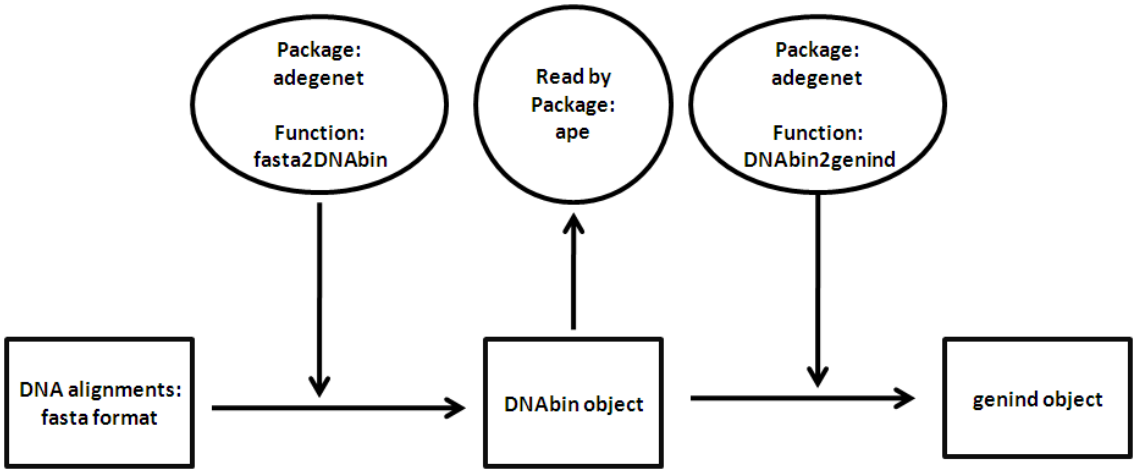


شکل ۹-۹ – پراکنش جغرافیایی دو جمیعت شبیه‌سازی شده در مجموعه داده sim2pop با درج مرز ژنتیکی و تمایز افراد متعلق به جمعیت‌ها با رنگ‌های مختلف

فصل دهم- داده‌های توالی

توالی‌های نوکلئوتیدی

توالی‌های نوکلئوتیدی را می‌توان با استفاده از پکیج ape در قالب اشیائی با کلاس DNAbin ذخیره نمود. تابع fasta2DNAbin در پکیج adegenet، خوانش همردیفی‌های دارای فرمت fasta (با پسوند fasta.، fas. یا fa.) و تبدیل به اشیاء DNAbin را، با حداقل میزان RAM لازم، امکان‌پذیر می‌سازد. از سوی دیگر اشیاء DNAbin با استفاده از تابع DNAbin2genind در پکیج adegenet به اشیاء genind قابل تبدیل می‌باشند. لذا با استفاده از توابع موجود در پکیج‌های مذکور می‌توان به راحتی همردیفی‌ها را خوانش کرد و داده‌ها را در نهایت در قالب اشیاء genind ذخیره نمود (شکل ۱۰-۱):



شکل ۱۰-۱ - نمودار راهنما برای خوانش و تبدیل داده‌های توالی DNA توسط پکیج‌های مختلف

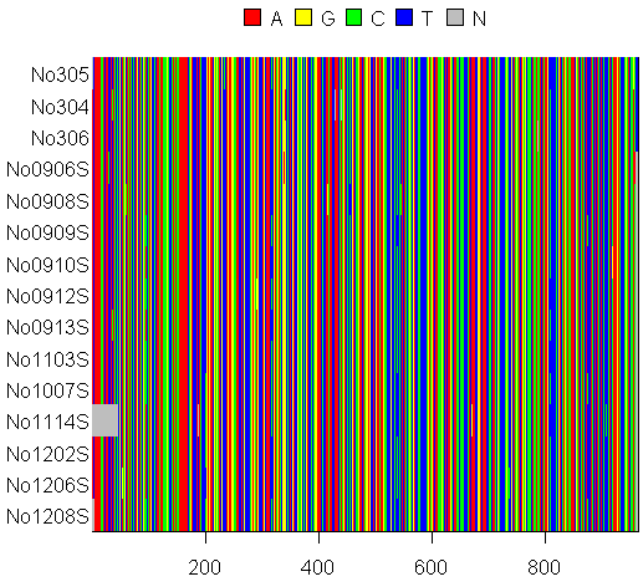
در اینجا برای آشنایی با اشیاء DNAbin، مجموعه داده woodmouse موجود در پکیج ape را مورد بررسی قرار می‌دهیم. این داده‌ها شامل ۱۵ توالی ژن میتوکندریایی سیتوکروم b در گونه‌ای از موش (*Apodemus*

sylvaticus) است که در اروپا و شمال غرب آفریقا زیست می‌کند و زیرمجموعه‌ای از داده‌های تجزیه شده توسط میچاکس و همکاران (Michaux *et al.*, 2003) می‌باشد.

```
> library(ape)
> data(woodmouse)
> class(woodmouse)
[1] "DNAbin"
> woodmouse
15 DNA sequences in binary format stored in a matrix.
All sequences of same length: 965
Labels: No305 No304 No306 No0906S No0908S No0909S ...
Base composition:
    a     c     g     t
0.307 0.261 0.126 0.306
```

با استفاده از تابع image، می‌توان نموداری از همردیفی توالی‌های شیء DNAbin ایجاد کرد (شکل ۱۰-۲):

```
> image(woodmouse)
```



شکل ۱۰-۲ – نمایش ترکیب بازی در همردیفی توالی‌های DNA موجود در مجموعه داده woodmouse

با استفاده از علامت کروشه [] می‌توان زیرمجموعه‌ای از داده‌های DNAbin را مشاهده یا در قالب شیء DNAbin به طور جداگانه ذخیره کرد. به عنوان مثال برای جدا کردن نوکلئوتیدهای ۲۰۱ تا ۴۵۰ از توالی‌های چهارم تا هفتم می‌توان دستور زیر را اجرا کرد:

```
> x <- woodmouse[4:7, 201:450]
> x
        4 DNA sequences in binary format stored in a matrix.
        All sequences of same length: 250
        Labels: No0906S No0908S No0909S No0910S
        Base composition:
          a     c     g     t
        0.318 0.181 0.174 0.327
```

تابع findMutations از پکیج adegenet، جهش‌های موجود بین دو توالی DNA همردیف شده را استخراج می‌نماید. این تابع به عنوان ورودی اشیاء DNAbin را می‌پذیرد.

مثال:

```
> library(ape)
> library(adegenet)
> data(woodmouse)
> findMutations(woodmouse[1:3,])
        $`No305->No304`
            No305 No304 short
        35    g     a     35:g->a
        36    t     c     36:t->c
        106   g     a     106:g->a
        ....
```

در مثال فوق جهش‌ها با در نظر گرفتن سه توالی اول در مجموعه داده‌های woodmouse استخراج شده است. توجه کنید که با توجه به نتایج فوق، اولین جهش در جایگاه نوکلئوتیدی ۳۵ مشاهده می‌شود. صحت این موضوع را می‌توان با بررسی وجود SNPها در مجموعه داده‌های woodmouse بررسی کرد:

```
> x <- DNAbin2genind(woodmouse)
```

```
> tab(x)[1:7,1:7]
```

	30.t	30.c	33.c	33.t	35.g	35.a	36.t
No305	1	0	1	0	1	0	1
No304	1	0	1	0	0	1	0
No306	1	0	1	0	0	1	1
No0906S	0	1	1	0	0	1	1
No0908S	1	0	1	0	0	1	1
No0909S	1	0	1	0	0	1	1
No0910S	1	0	1	0	0	1	1

همانطور که مشاهده می‌شود سه توالی اول، در جایگاه ۳۰ و ۳۳ مشابه می‌باشند و اولین جایگاه متفاوت در آنها شماره ۳۵ است.

همچنین می‌توان با استفاده از تابع as.character جایگاه‌های نوکلئوتیدی (مثلاً جایگاه شماره ۳۵، ۳۶ و ۱۰۶ در مثال فوق) را بین توالی‌ها مورد بررسی قرار داد:

```
> as.character(woodmouse)[1:3,35]
    No305 No304 No306
     "g"   "a"   "a"
> as.character(woodmouse)[1:3,36]
    No305 No304 No306
     "t"   "c"   "t"
> as.character(woodmouse)[1:3,106]
    No305 No304 No306
     "g"   "a"   "a"
```

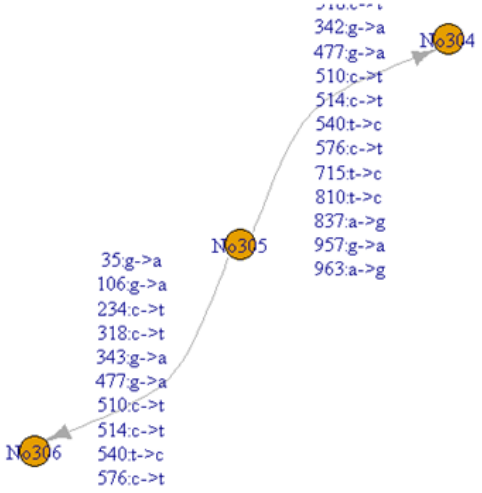
از آرگومان‌های from و to می‌توان برای تعیین اینکه جهش‌ها از کدام توالی (from) به کدام توالی (to) خوانده شوند، استفاده کرد. به تفاوت خروجی دو دستور زیر توجه کنید:

```
> findMutations(woodmouse[1:3,], from=1)
$`No305->No304`
     No305 No304 short
35       g     a 35:g->a
36       t     c 36:t->c
106      g     a 106:g->a
234      c     t 234:c->t
318      c     t 318:c->t
.....
```

```
> findMutations(woodmouse[1:3,], from=2)

$`No304->No305`
     No304 No305 short
35       a     g 35:a->g
36       c     t 36:c->t
106      a     g 106:a->g
234      t     c 234:t->c
318      t     c 318:t->c

……..…
```

تابع graphMutations عملکرد مشابهی دارد، علاوه بر اینکه نموداری از جهش‌های استخراج شده ایجاد می‌نماید (شکل ۱۰-۳):

```
> g <- graphMutations(woodmouse[1:3,], from=1)
```

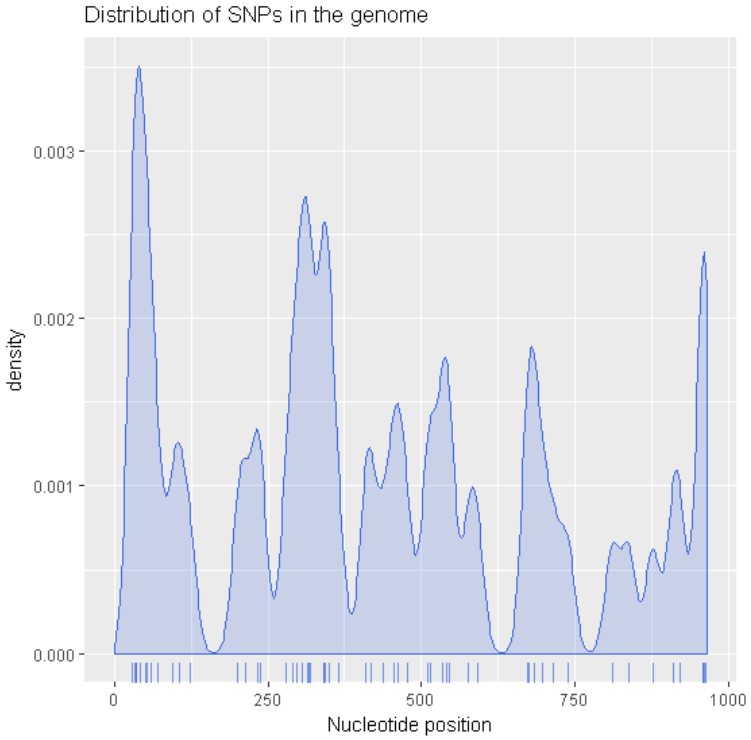


شکل ۱۰-۳ – نمایش جهش‌های استخراج شده از همردیفی توالی‌های DNA موجود در مجموعه داده woodmouse

با استفاده از تابع snpposi.plot در پکیج adegenet، می‌توان موقعیت و تراکم SNPها را در یک همردیفی ترسیم نمود (شکل ۱۰-۴). همچنین با استفاده از تابع snpposi.test می‌توان توزیع تصادفی SNPها یا تجمع خوشه‌ای آنها را در ژنوم، آزمون کرد. این آزمون براساس فاصله هر SNP نسبت به نزدیکترین SNPها می‌باشد. بدین ترتیب معیاری از خوشه‌بندی برای هر SNP حاصل می‌شود. با استفاده از آرگومان stat، آماره‌های مختلفی را می‌توان بدین منظور بکار برد، اما آماره پیش فرض در این رابطه، عبارت از میانه

(median) می‌باشد. این توابع بر روی اشیاء DNAbin قابل کاربرد می‌باشند. به عنوان مثال، توزیع SNPها را در مجموعه داده woodmouse، مورد بررسی قرار می‌دهیم:

```
> library(adegenet)
> library(ape)
> data(woodmouse)
> snpposi.plot(woodmouse, codon=FALSE)
```



شکل ۱۰-۴ – توزیع SNPهای استخراج شده از همردیفی توالی‌های DNA موجود در مجموعه داده woodmouse

با انتخاب آرگومان codon=TRUE می‌توان موقعیت کدون را در نمودار توزیع SNPها مشخص نمود (شکل ۱۰-۵). توجه داشته باشید که این آرگومان بطور پیش فرض دارای مقدار TRUE می‌باشد و لذا عدم درج آن در تابع نیز نتیجه مشابهی خواهد داشت.

```
snpposi.plot(woodmouse, codon=TRUE)
```

شکل ۱۰-۵ – توزیع SNPهای استخراج شده از همردیفی توالی‌های DNA موجود در مجموعه داده woodmouse به تفکیک موقعیت کدون

آزمون توزیع تصادفی SNPها در ژنوم با استفاده از تابع snpposi.test، برای مجموعه داده woodmouse، بصورت زیر قابل انجام می‌باشد:

```
> library(adegenet)
> library(ape)
> data(woodmouse)
> snpposi.test(woodmouse)

        Monte-Carlo test
        Call: as.randtest(sim = sim, obs = obs, alter = "less")
        Observation: 6
        Based on 999 replicates
        Simulated p-value: 0.511
        Alternative hypothesis: less


     Std.Obs Expectation    Variance
```

-0.3351054 6.4479479 1.7868670

براساس نتایج آزمون فوق، مقدار بزرگ p-value، توزیع تصادفی SNPها در ژنوم، و در نتیجه، عدم تجمع خوشه‌ای آنها را نشان می‌دهد.

انتساب جمعیت به داده‌های توالی

برای اینکه بتوان توابع تجزیه نشانگرهای ژنتیکی را بر روی داده‌های توالی اعمال کرد لازم است ابتدا اشیاء DNAbin با استفاده از تابع DNAbin2genind به اشیاء gnind تبدیل شوند. در اینصورت انتساب جمعیت‌های منشاء توالی‌ها (درصورتی که توالی‌ها از جمعیت‌های متفاوتی گرفته شده باشند) نیز امکان‌پذیر خواهد بود. از آنجا که برای مجموعه توالی‌های woodmouse، جمعیتی مشخص نشده است، در اینجا به عنوان مثال جمعیت‌هایی را بصورت فرضی و به طور تصادفی، تعریف می‌نماییم و آماره‌های ژنتیکی را براساس این جمعیت‌های فرضی برآورد و آزمون می‌نماییم.

```
> library(adegenet)
> library(ape)
> data(woodmouse)
> wmg<-DNAbin2genind(woodmouse)
> wmg
        /// GENIND OBJECT /////////
         // 15 individuals; 56 loci; 114 alleles; size: 25.3 Kb
         // Basic content
           @tab:  15 x 114 matrix of allele counts
           @loc.n.all: number of alleles per locus (range: 2-3)
           @loc.fac: locus factor for the 114 columns of @tab
           @all.names: list of allele names for each locus
           @ploidy: ploidy of each individual  (range: 1-1)
           @type:  codom
           @call: DNAbin2genind(x = woodmouse)
         // Optional content
```

```
- empty -
> a<-rep(LETTERS[1:3],5)
> a
 [1] "A" "B" "C" "A" "B" "C" "A" "B" "C" "A" "B" "C" "A" "B" "C"
> set.seed(20170615)
> b<-sample(a,15)
> b
 [1] "B" "C" "A" "C" "B" "A" "A" "B" "B" "C" "A" "B" "A" "C" "C"
> pop(wmg) <-b
> pop(wmg)
 [1] B C A C B A A B B C A B A C C
Levels: B C A
```

آماره‌های افتراق ژنتیکی جمعیت

از آنجا که شیء ایجاد شده از تابع DNAbin2genind از نوع gnind است، همان توابع برآورد آماره‌های افتراق ژنتیکیِ جمعیت که در بخش‌های پیشین توضیح داده شد را در اینجا می‌توان مورد استفاده قرار داد، لذا جهت پرهیز از اطاله کلام فقط به ذکر مثال بسنده کرده و از توضیح مجدد آنها خودداری می‌نماییم. خوانندگان جهت اطلاعات بیشتر می‌توانند به بخش‌های مذکور مراجعه نمایند. در اینجا به عنوان مثال آماره‌های افتراق ژنتیکی را برای شیء wmg که در بخش پیش ایجاد شد، به دست می‌آوریم:

```
> library(mmod)
> diff_stats(wmg)
$per.locus
```

	Hs	Ht	Gst	Gprime_st	D
30	0.119626	0.128764	0.070968	0.116773	0.01557
33	0.119626	0.128764	0.070968	0.116773	0.01557
35	0.140187	0.15784	0.111842	0.184783	0.030797
36	0.259813	0.264382	0.017282	0.034722	0.009259
…….					
920	0.118519	0.128395	0.076923	0.12605	0.016807

957	0.118519	0.128395	0.076923	0.12605	0.016807
959	0.118519	0.128395	0.076923	0.12605	0.016807
960	0.118519	0.128395	0.076923	0.12605	0.016807
963	0	0.444444	1	1	0.666667

$global

Hs	Ht	Gst_est	Gprime_st	D_het	D_mean
0.204557	0.22509	0.091224	0.164521	0.038721	NA

به منظور برآورد آماره Gst نی (Nei, 1973) به صورت دو به دو بین جمعیت‌ها، می‌توان از تابع pairwise_Gst_Nei در پکیج mmod استفاده نمود:

```
> wmg.pg<-pairwise_Gst_Nei(wmg, linearized = FALSE)
> wmg.pg
```

	B	C
C	0.071297	
A	0.076185	0.061416

باید توجه داشت که اگر برای آرگومان linearized گزینه TRUE انتخاب شود، برآورد خطی Gst، یعنی Gst(1/(1-Gst)) ارائه می‌شود. خروجی تابع فوق یک ماتریس فاصله می‌باشد:

```
> class(wmg.pg)
[1] "dist"
```

ترسیم فواصل ژنتیکی

همانطور که در بخش‌های پیشین توضیح داده شد با استفاده از تابع gengraph در پکیج adegenet می‌توان نموداری براساس فواصل ژنتیکی بین افراد یا جمعیت‌ها با تعیین یک حد آستانه ترسیم نمود. تابع gengraph، اشیاء DNAbin را نیز به عنوان ورودی می‌پذیرد. به عنوان مثال در مجموعه داده‌های woodmouse ، فواصل ژنتیکی را بصورت زیر رسم می‌نماییم:

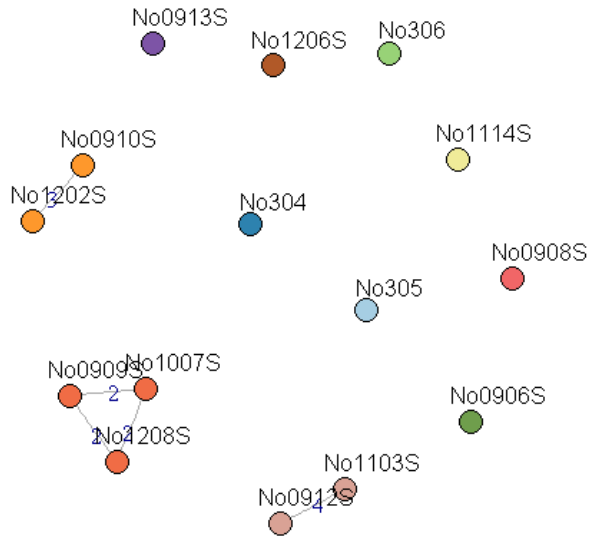
```
> library (ape)
> library (adegenet)
> data(woodmouse)
> g <- gengraph(woodmouse, cutoff=5)
> g
```

```
$graph
IGRAPH UNW- 15 5 --
+ attr: name (v/c), color (v/c), label (v/c), label.dist (v/n), size
| (v/n), label.family (v/c), label.color (v/c), weight (e/n), label
| (e/n)
+ edges (vertex names):
[1] No0909S--No1007S No0909S--No1208S No0910S--No1202S No0912S--No1103S
[5] No1007S--No1208S

$clust
$clust$membership
 No305  No304  No306 No0906S No0908S No0909S No0910S No0912S No0913S No1103S
     1      2      3      4      5      6      7      8      9      8
No1007S No1114S No1202S No1206S No1208S
     6      10      7      11      6

$clust$csize
 [1] 1 1 1 1 1 3 2 2 1 1 1

$clust$no
[1] 11


$cutoff
[1] 5

$col
      1       2       3       4       5       6       7       8
#A6CEE3 #2D82AF #98D277 #6F9E4C #F16667 #F06C45 #FE982C #D9A295
      9      10      11
#7D54A6 #F0EB99 #B15928
```

همانطور که مشاهده می‌شود خروجی تابع gengraph، لیستی متشکل از اجزاء مختلف است. $clust حاوی اطلاعات مربوط به کلاسترها است که در آن، جزء $membership، عضویت موجودیت‌ها در کلاسترهای گوناگون، $csize، اندازه کلاسترها و $no تعداد کلاسترها را مشخص می‌کند.

با استفاده از جزء $graph، می‌توان نمودار پراکنش موجودیت‌ها را ترسیم نمود (شکل ۱۰-۶):

```
> plot(g$graph)
```



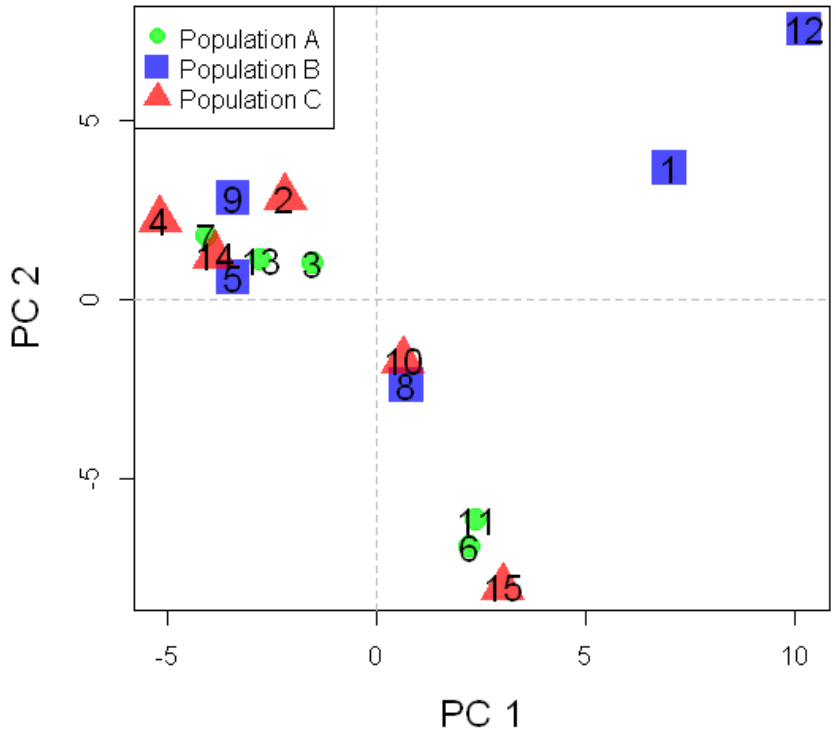
شکل ۱۰-۶ – توزیع توالی‌های DNA موجود در مجموعه داده woodmouse برحسب فواصل ژنتیکی

تجزیه به مؤلفه‌های اصلی (PCA)

به طور مشابه با بخش پیشین، با تبدیل داده‌های توالی از کلاس DNAbin به اشیاء genind، توابع مربوط به تجزیه به مؤلفه‌های اصلی نیز بر روی آنها، قابل استفاده خواهند بود. لذا به عنوان مثال این تجزیه را برای داده‌های توالی مجموعه woodmouse که توسط تابع DNAbin2genind به شیء genind تبدیل شده (ادامه از بخش‌های قبل) انجام می‌دهیم (شکل ۱۰-۷):

```
> s.wmg<- scaleGen(wmg, NA.method="mean")
> wmg.pca <- dudi.pca(s.wmg,scannf=FALSE,scale=FALSE, nf=2)
> temp <- as.integer(pop(wmg))
> my.col <- transp(c("blue","red", "green"),.7)[temp]
> my.pch <- c(15,17,20)[temp]
> indNames(wmg)=1:nInd(wmg)
> nme=indNames(wmg)
> plot(wmg.pca$li, xlab="PC 1", ylab="PC 2", cex.lab=1.5, col=my.col, cex=3, pch=my.pch)
> text(wmg.pca$li[,1], wmg.pca$li[,2], nme, cex=1.4)
```

```
> abline(h=0,v=0,col="grey",lty=2)

> legend("topleft", pch=c(20,15,17), col=transp(c("green","blue","red"),.7), leg=c("Population A","Population B", "Population C"), pt.cex=2)
```



شکل ۱۰-۷ – تفکیک توالی‌های DNA موجود در مجموعه داده woodmouse در نمودار مبتنی بر مؤلفه‌های اصلی

همانطو که در نمودار پراکنش مشاهده می‌شود، توالی‌های شماره ۱ و ۱۲ (از جمعیت B) براساس بر دو مؤلفه اصلی اول از سایر توالی‌های متمایز شده‌اند، اما مؤلفه‌های اصلی در جداسازی سه جمعیت از یکدیگر موفق عمل نکرده است که با توجه به اینکه جمعیت‌های مذکور بصورت فرضی، تعریف و به صورت تصادفی منتسب شده بودند، نتیجه‌ای مورد انتظار می‌باشد.

تجزیه واریانس مولکولی (AMOVA)

با تبدیل داده‌های توالی مجموعه woodmouse به شیء genind، تجزیه واریانس مولکولی نیز توسط تابع amova از پکیج pegas بر روی آن قابل اعمال است. همانطور که قبلاً توضیح داده شد برای انجام تجزیه AMOVA، باید نوعی دسته‌بندی در ژنوتیپ‌ها برای تسهیم واریانس ژنتیکی بین‌گروهی و درون‌گروهی، موجود باشد. در اینجا همان جمعیت‌های فرضی که در بخش‌های پیشین تعریف شده به عنوان معیار دسته‌بندی قرار می‌گیرد (ادامه از بخش‌های پیشین):

```
> my.stra<-data.frame(Popu=b)
> strata(wmg) <- my.stra
> my.dist<-dist.dna(woodmouse)
> wmg_amova <-pegas::amova(my.dist ~Popu, data = my.stra, nperm =10)
> wmg_amova
```

Analysis of Molecular Variance

Call: pegas::amova(formula = my.dist ~ Popu, data = my.stra, nperm = 10)

	SSD	MSD	df
Popu	0.000157	7.86E-05	2
Error	0.001202	1.00E-04	12
Total	0.00136	9.71E-05	14

Variance components:

	sigma2	P.value
Popu	-4.32E-06	0.6364
Error	1.00E-04	

Variance coefficients:

a

5

مقدار عددیP.value بزرگ بوده که بیانگر عدم معنی‌دار بودن تفاوت بین جمعیت‌ها است که با توجه به اینکه جمعیت‌های مذکور بصورت فرضی، تعریف و بصورت تصادفی منتسب شده بودند، نتیجه‌ای مورد انتظار می‌باشد.

توجه داشته باشید که در دستورات فوق، از تابع dist.dna برای محاسبه فاصله ژنتیکی بین توالی‌ها استفاده شده است که تابعی از پکیج ape می‌باشد. تابع dist.dna برای محاسبه فاصله بین جفت توالی‌های DNA براساس یازده مدل مختلف، قابل کاربرد است. مدل کیمورا (Kimura,1980) به عنوان پیش فرض در تابع dist.dna، برای محاسبه فواصل بین جفت توالی‌های DNA در نظر گرفته شده است. برای استفاده از سایر مدل‌ها باید از آرگومان model در تابع dist.dna، استفاده نمود. به عنوان مثال برای مدل فلسنستین (Felsenstein,1981) از آرگومان model="F8" استفاده می‌شود:

```
> my.dist<-dist.dna(woodmouse, model="F81")
> ma.dist<-as.matrix(my.dist)
> ma.dist[1:5,1:5]
```

	No305	No304	No306	No0906S	No0908S
No305	0.000000	0.014428	0.013308	0.017798	0.016673
No304	0.014428	0.000000	0.003304	0.012189	0.011073
No306	0.013308	0.003304	0.000000	0.008845	0.007733
No0906S	0.017798	0.012189	0.008845	0.000000	0.012189
No0908S	0.016673	0.011073	0.007733	0.012189	0.000000

همچنین می‌توان از گزینه‌های JC69، K81، F84، BH87، T92، GG95، logdet و paralin به ترتیب برای مدل‌های جوکس و کانتور (Jukes and Cantor,1969)، کیمورا (Kimura, 1981)، فلسنستین (Felsenstein, 1984)، بری و هارتیگان (Barry and Hartigan, 1987)، تامورا و نی (Tamura and Nei, 1993)، گالتیر و گوی (Galtier and Gouy, 1995)، لاکهرت و همکاران (Lockhart *et al.*, 1994) و لیک (Lake, 1994) استفاده نمود. گزینه‌های raw، N، TS، indel TV و indelblock نیز بدین منظور قابل استفاده می‌باشد. برای توضیح در مورد این گزینه‌ها می‌توان دستور زیر را اجرا و محتویات صفحه مرورگری که در پی اجرای دستور باز می‌شود را، مطالعه نمود:

```
> help(dist.dna)
```

توالی‌های اسید آمینه

توالی‌های اسیدآمینه موجود در یک همردیفی را می‌توان بازخوانی نموده و برای شناسایی جایگاه‌های پلی‌مورفیک مورد تجزیه قرار داد. تابع read.alignment در پکیج seqinr، همردیفی‌های متشکل از توالی اسیدآمینه‌ای را خوانش کرده و در قالب شیئی از نوع alignment در می‌آورد. اشیاء alignment با استفاده از تابع alignment2genind در پکیج adegenet قابل تبدیل به اشیاء genind می‌باشند. لذا پس از این مرحله، تمام انواع تجزیه که بر روی اشیاء genind قابل انجام است، برای همردیفی توالی‌های اسیدآمینه که به شیوه‌ی فوق‌الذکر به اشیاء genind تبدیل شده‌اند، قابل انجام خواهد بود. در اینجا برای تمرین از فایلی بنام test.mase که هنگام دانلود پکیج seqinr در پوشه library/seqinr/sequences/test.mase نصب می‌شود، استفاده می‌نماییم. مسیر دسترسی به این فایل با دستور زیر قابل تشخیص است:

```
> library(seqinr)
> system.file("sequences/test.mase", package="seqinr")
    [1] "D:/Program Files/R/R-3.2.2/library/seqinr/sequences/test.mase"
```

با استفاده از تابع read.alignment در پکیج seqinr، همردیفی موجود در فایل فوق را خوانش می‌نماییم:

```
> test.alig <- read.alignment(file=system.file("sequences/test.mase", package="seqinr"), format = "mase")
> class(test.alig)
    [1] "alignment"
```

همانطور که مشاهده می‌شود خروجی تابع read.alignment، شیئی از نوع alignment می‌باشد. این شیء از اجزاء مختلفی از جمله توالی‌های موجود در همردیفی، تشکیل شده است که با علامت $ قابل فراخوانی می‌باشند:

```
> test.alig
    $nb
    [1] 6

    $nam
    [1] "Langur" "Baboon" "Human"  "Rat"    "Cow"    "Horse"

    $seq
```

$seq[[1]]

[1] "-
kifercelartlkklgldgykgvslanwvclakwesgynteatnynpgdestdygifqinsrywcnngkpgavdachiscsallqnniadavac
akrvvsdqgirawvawrnhcqnkdvsqyvkgcgv-"

$seq[[2]]

[1] "-
kifercelartlkrlgldgyrgislanwvclakwesdyntqatnynpgdqstdygifqinshywcndgkpgavnachiscnallqdnitdavaca
krvvsdqgirawvawrnhcqnrdvsqyvqgcgv-"

$seq[[3]]

[1] "-
kvfercelartlkrlgmdgyrgislanwmclakwesgyntratnynagdrstdygifqinsrywcndgkpgavnachlscsallqdniadavac
akrvvrdqgirawvawrnrcqnrdvrqyvqgcgv-"

$seq[[4]]

[1] "-
ktyercefartlkrngmsgyygvsladwvclaqhesnyntqarnydpgdqstdygifqinsrywcndgkpraknacgipcsallqdditqaiqc
akrvvrdqgirawvawqrhcknrdlsgyirncgv-"

$seq[[5]]

[1] "-
kvfercelartlkklgldgykgvslanwlcltkwessyntkatnynpssestdygifqinskwwcndgkpnavdgchvscselmendiakava
cakkivseqgitawvawkshcrdhdvssyvegctl-"

$seq[[6]]

[1] "-
kvfskcelahklkaqemdgfggyslanwvcmaeyesnfntrafngknangssdyglfqlnnkwwckdnkrsssnacnimcskllldenidd
discakrvvrdkgmsawkawvkhckdkdlseylascnl-"

$com

[1] ";empty description\n" ";\n" ";\n"

[4] ";\n" ";\n" ";\n"

```
attr(,"class")
[1] "alignment"
```

با استفاده از تابع alignment2genind در پکیج adegenet، شیء فوق را به شیء genind تبدیل می‌نماییم:

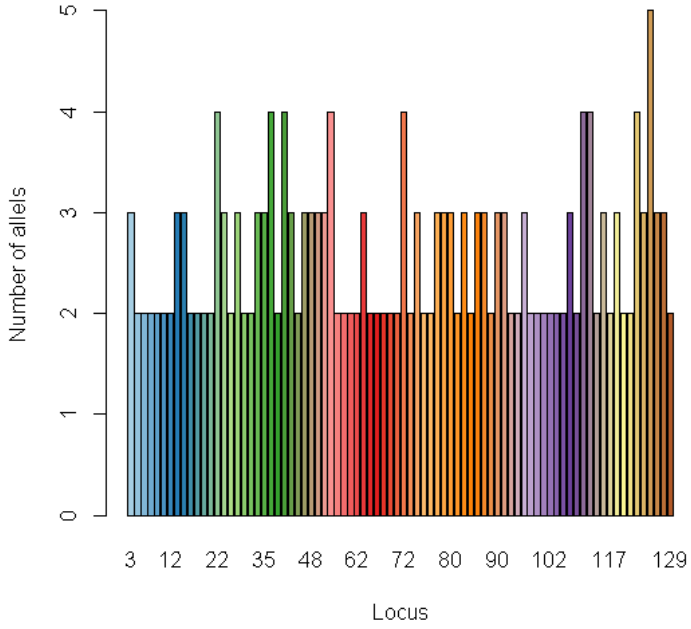
```
> library(adegenet)
> test.geni <- alignment2genind(test.alig)
> test.geni
/// GENIND OBJECT /////////
 // 6 individuals; 82 loci; 212 alleles; size: 34.4 Kb
 // Basic content
   @tab:  6 x 212 matrix of allele counts
   @loc.n.all: number of alleles per locus (range: 2-5)
   @loc.fac: locus factor for the 212 columns of @tab
   @all.names: list of allele names for each locus
   @ploidy: ploidy of each individual  (range: 1-1)
   @type:  codom
   @call: alignment2genind(x = test.alig)
 // Optional content
   @other: a list containing: com
```

همانطور که مشاهده می‌شود ۸۲ جایگاه پلی‌مورفیک از همردیفی توالی‌های اسیدآمینه استخراج شده که با در نظر گرفتن باقی‌مانده‌های (اسیدآمینه‌های) متفاوت در هرجایگاه پلی‌مورفیک، در مجموع ۲۱۲ آلل شناسایی شده است. اسم جایگاه پلی‌مورفیک با همان عدد مربوط به محل قرار گرفتن جایگاه پلی‌مورفیک مشخص می‌شود:

```
> head(locNames(test.geni))
[1] "3"  "4"  "5"  "6"  "9"  "11"
> length(locNames(test.geni))
[1] 82
```

با ترسیم نمودار ستونی مربوط به فراوانی آلل‌های هر یک از جایگاه‌های پلی‌مورفیک، می‌توان تصویری از توزیع پلی‌مورفیسم در طول همردیفی بدست آورد (شکل ۱۰-۸):

```
> barplot(test.geni @loc.n.all, col=funky(82), xlab="Locus", ylab="Number of allels")
```



شکل ۱۰-۸ –فراوانی آلل‌ها در جایگاه‌های پلی‌مورفیک استخراج شده از همردیفی توالی‌های اسیدآمینه موجود در مجموعه داده test

با تبدیل شیء genind به قالب داده، می‌توان جدولی متشکل از اسیدآمینه‌های موجود در جایگاه‌های پلی‌مورفیک به تفکیک توالی‌ها، تشکیل داد:

```
> test.df <- genind2df(test.geni)
> dim(test.df)
[1]  6 82
> test.df[, 1:20]
```

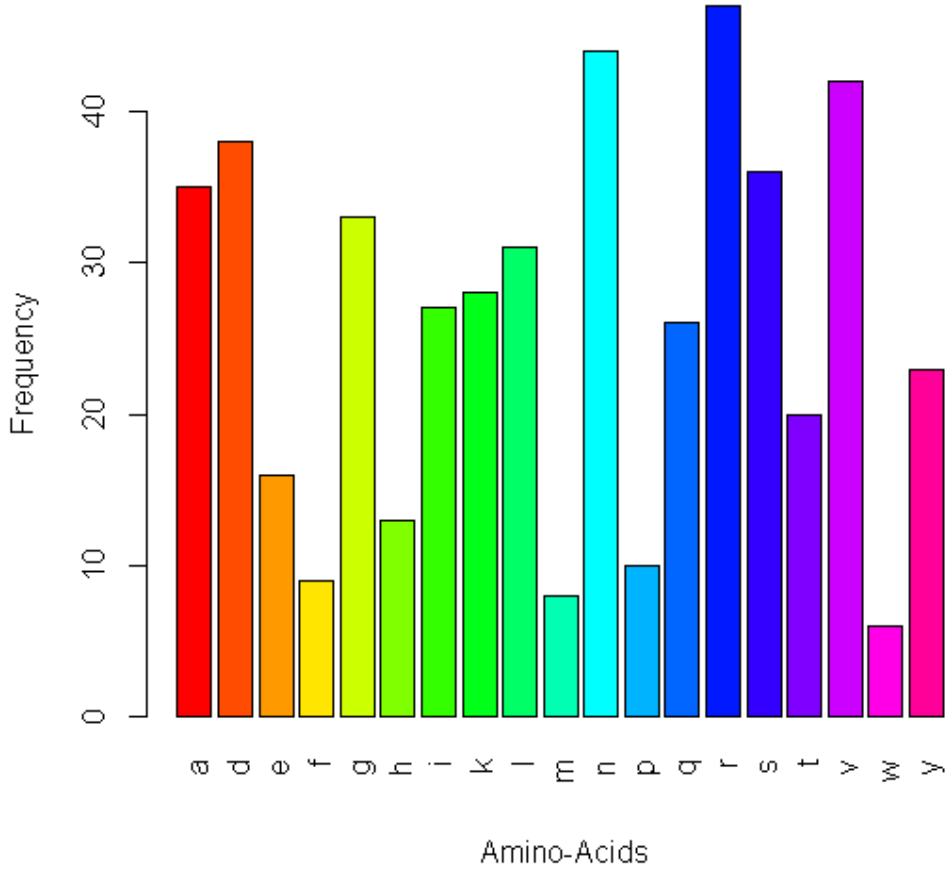
	3	4	5	6	9	11	12	15	16	17	18	19	21	22	24	28	30	32	33	34
Langur	i	f	e	r	l	r	t	k	l	g	l	d	y	k	v	n	v	l	a	k
Baboon	i	f	e	r	l	r	t	r	l	g	l	d	y	r	i	n	v	l	a	k
Human	v	f	e	r	l	r	t	r	l	g	m	d	y	r	i	n	m	l	a	k
Rat	t	y	e	r	f	r	t	r	n	g	m	s	y	y	v	d	v	l	a	q
Cow	v	f	e	r	l	r	t	k	l	g	l	d	y	k	v	n	l	l	t	k
Horse	v	f	s	k	l	h	k	a	q	e	m	d	f	g	y	n	v	m	a	e

همچنین می‌توان با دستورات زیر، ترکیب اسیدهای آمینه در جایگاه‌های پلی‌مورفیک را بدست آورد و براساس آن نمودار ستونی ترسیم نمود (شکل ۱۰-۹):

```
> test.tab<-table(unlist(test.df))
> test.tab
```

a	d	e	f	g	h	i	k	l	m	n	p	q	r	s	t	v	w	y
35	38	16	9	33	13	27	28	31	8	44	10	26	47	36	20	42	6	23

```
> barplot(test.tab, col=rainbow(20), las=3,xlab=" Amino-Acids", ylab="Frequency")
```



شکل ۱۰-۹ – فراوانی انواع اسید آمینه در جایگاه‌های پلی‌مورفیک استخراج شده از همردیفی توالی‌های موجود در مجموعه داده test

فواصل ژنتیکی بین توالی‌ها

پس از تبدیل شیء alignmnet مربوط به همردیفی توالی‌های اسیدآمینه به شیء genind به شرح فوق‌الذکر، می‌توان با استفاده از تابع dist، فاصله اقلیدسی بین توالی‌ها را محاسبه نمود. باید توجه داشت که این تابع، فواصل ساده اقلیدسی را محاسبه می‌نماید و برای فواصل ویژه‌ی توالی‌های اسیدآمینه، بایستی از توابع اختصاصی مربوطه در سایر پکیج‌ها استفاده نمود که به عنوان مثال می‌توان از تابع dist.alignment در پکیج seqinr یا از تابع dist.ml در پکیج phangorn نام برد. در اینجا فاصله اقلیدسی بین توالی‌ها در شیء test.geni که در بخش قبل ایجاد شد را محاسبه می‌نماییم:

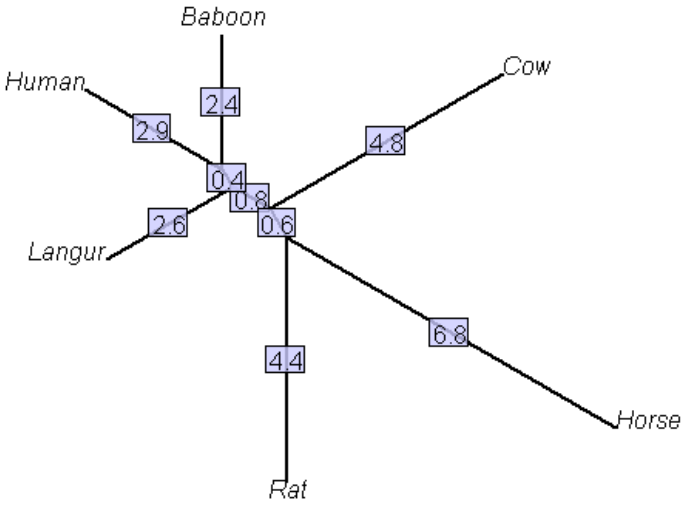
```
> test.dist <- dist(tab(test.geni))
> test.dist
```

	Langur	Baboon	Human	Rat	Cow
Baboon	5.291503				
Human	6	5.291503			
Rat	8.717798	8.124038	8.602325		
Cow	7.874008	8.717798	8.944272	10.39231	
Horse	11.31371	11.31371	11.22497	11.22497	11.74734

ترسیم درخت فیلوژنی

با محاسبه فواصل ژنتیکی بین توالی‌ها، ترسیم درخت فیلوژنی توسط توابعی که بدین منظور توسعه یافته‌اند آسان خواهد بود. به عنوان مثال در اینجا از تابع nj در پکیج ape برای ترسیم درخت فیلوژنی به روش اتصال همسایه‌ها (شکل ۱۰-۱۰) استفاده می‌نماییم (ادامه دستورات از بخش قبل):

```
> library(ape)
> test.tree <- nj(test.dist)
> opar<-par()
> par(xpd=TRUE)
> plot(test.tree, type="unrooted", edge.w=2)
> edgelabels(tex=round(test.tree$edge.length,1), bg=rgb(.8,.8,1,.8))
> par(opar)
```

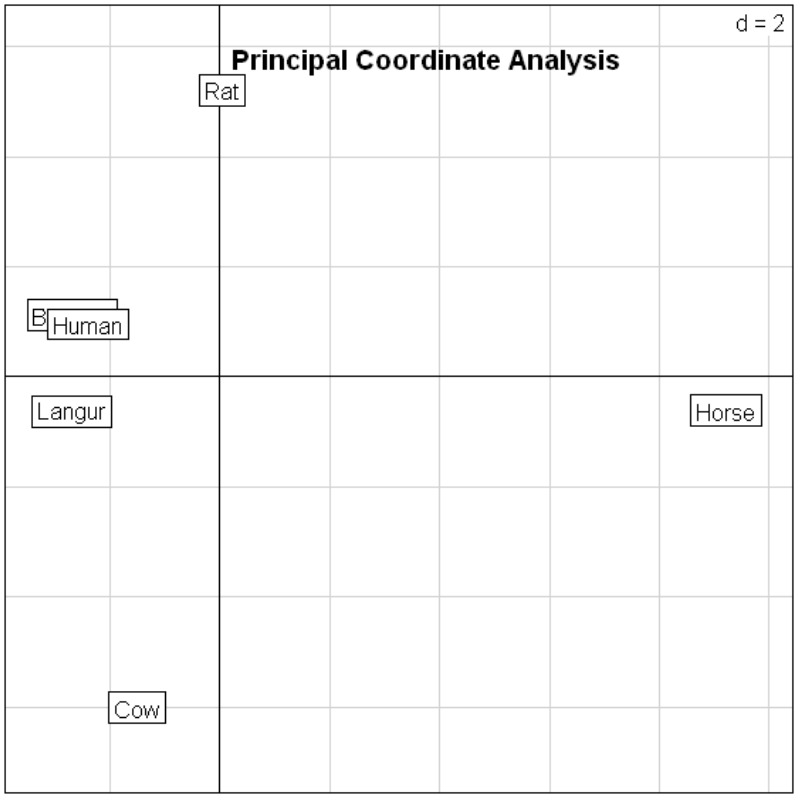


شکل ۱۰-۱۰ – درخت فیلوژنی برای توالی‌های اسید آمینه موجود در مجموعه داده test

تجزیه به مختصات اصلی

براساس فواصل ژنتیکی محاسبه شده بین توالی‌ها می‌توان ابعاد داده‌ها در ۲۱۲ متغیر مربوط به آلل‌ها را با استفاده از روش تجزیه به مختصات اصلی کاهش، و توزیع اعضاء را در نموداری مبتنی بر محورهای مختصات اصلی (شکل ۱۰-۱۱) نمایش داد (ادامه دستورات از بخش قبل):

```
> test.pco <- dudi.pco(test.dist, scannf=FALSE,nf=2)
> s.label(test.pco$li*1.1)
> title("Principal Coordinate Analysis")
```



شکل ۱۰-۱۱ – تفکیک توالی‌های اسید آمینه موجود در مجموعه داده test در نمودار مبتنی بر محورهای تجزیه به مختصات اصلی

فصل یازدهم-داده‌های انبوه (اشیاء genlight)

مقدمه

با ظهور نسل جدید فناوری‌های توالی‌یابی، داده‌های ژنوم کامل موجودات به طور گسترده‌ای در دسترس قرار گرفته است. بدین ترتیب حجم عظیم از داده‌های ژنتیکی در حال ایجاد می‌باشد که تفاوت ژنوم موجودات در حد تک نوکلئوتید، که تحت عنوان شناخته SNP می‌شود، را در سطح ژنوم آشکار می‌سازند. بدین ترتیب انبوه داده‌های متشکل از SNPها به عنوان نشانگر ژنتیکی، توجهات را جلب نمود. SNPها فراوان‌ترین و قدرتمندترین نشانگر به منظور انجام ژنوتیپ‌یابی پربازده می‌باشند (Gunderson 2007; Steemers and Alkan *et al*. 2011) و در حال حاضر به عنوان نوعی نشانگر DNA که در هر ژنی امکان حضور دارند، به شدت مورد توجه جامعه علمی قرار گرفته‌اند (Rafalski, 2002). نشانگرهای SNP، تنوع ژنومی گونه‌ها را مشخص می‌نمایند و با استفاده از آنها می‌توان گونه‌زایی و تکامل را نشان داد و ارتباط بین تنوع ژنومی و صفات فنوتیپی را بررسی نمود (McNally *et al*. 2009). همانطور که از نام این نشانگرها پیدا است، SNP عبارت از یک تغییر منفرد در توالی DNA در یک جایگاه مشخص می‌باشد. برای اینکه حالت‌های متناظر برای یک جایگاه بازی در DNA ژنومی به عنوان SNP در نظر گرفته شود بایستی کم‌وقوع‌ترین آلل، حداقل دارای فراوانی یک درصد یا بیشتر باشد. هرچند که از لحاظ نظری، در هر جایگاه، هر یک از چهار باز نوکلئوتیدی می‌توانند حضور داشته باشند، اما SNPها در عمل دو آللی می‌باشند. یکی از دلایل این موضوع فراوانی پایین جایگزینی‌های تک نوکلئوتیدی در پیدایش SNPها است که به عنوان مثال در پستانداران تخمین زده می‌شود بین ۱×۱۰^{-۹} و ۵×۱۰^{-۹}، در هر نوکلئوتید در هر سال، در جایگاه‌های خنثی باشد. بنابراین احتمال اینکه دو باز بطور مستقل در یک جایگاه تغییر کنند، بسیار پایین است. دلیل دیگر، عدم توازن در رخداد جهش است که سبب بوجود آمدن صرفاً دو نوع SNP می‌شود. می‌دانیم که بطور کلی دو نوع جهش وجود دارد که عبارت از جهش همجنس بین پورین‌ها (A⇔G) یا بین پیریمیدین‌ها (C⇔T) و

جهش ناهمجنس بین پورین‌ها و پیریمیدین‌ها (A⇔C، A⇔T، G⇔C و G⇔T) می‌باشد. بنابراین انواع حالات جهش‌های ناهمجنس دو برابر انواع جهش‌های همجنس است و لذا انتظار می‌رود چنانچه وقوع جهش تصادفی باشد نسبت فراوانی جهش همجنس به جهش ناهمجنس، ۰/۵ باشد، اما مشاهدات، انحراف آشکاری به سمت وقوع جهش همجنس نشان می‌دهند (Gojobori and Nei, 1981، Martínez-Arias *et al.*, 2001، Collins *et al.*, 1994، Picoult-Newberg *et al.*, 1999، Smith *et al.*, 2001، Kim *et al.*, 2002، Cooper and Krawczak, 1989 و Bray *et al.*, 2001).

اشیاء genlight و پردازش آنها

حجم عظیم داده‌های ژنتیکی که در نتیجه‌ی کاربرد فناوری‌های نوین تولید شده، ذخیره و مدیریت آنها را دچار چالش ساخته است. پکیج adegenet کلاس جدیدی از اشیاء (موسوم به کلاس genlight) را برای پردازش پلی‌مورفیسم (SNPها) در سراسر ژنوم با حداقل نیاز به RAM ایجاد نموده است. به منظور ذخیره داده‌های ژنوتیپی SNP به صورت دودوئی و به شیوه فشرده، از کلاس genlight استفاده می‌شود. این روش ذخیره‌سازی، خصوصاً برای داده‌های هاپلوئیدی بسیار کارآمد است و حافظه اختصاص یافته لازم را تا ۵۰ برابر کاهش می‌یابد. با اینحال از genlight می‌توان برای هر سطحی از پلوئیدی، استفاده نمود. تنها نقطه ضعف این سیستم آن است که فقط جایگاه‌های SNP دو آللی را حفظ می‌کند و در نتیجه مختصری اطلاعات مربوط به جایگاه‌های چند آللی، از دست می‌رود. یک شیء genlight متشکل از وکتورهایی از اعداد است که نشان‌دهنده‌ی تعدادِ دومین آلل، در هر لوکوس و هر فرد می‌باشد. genlight قادر است چندین ژنوتیپ را ذخیره کند. با استفاده از تابع new("genlight", x) می‌توان یک شیء genlight ایجاد کرد که در آن x، لیستی از تعداد آلل دوم مربوط به لوکوس‌های SNP است.

مثال: سه ژنوتیپ به نام‌های indiv1، indiv2 و indiv3، به شرح زیر تعریف می‌شود. سپس لیستی از آن‌ها تهیه شده و شیء genlight ایجاد می‌شود:

```
> library(adegenet)
> dat <- list(indiv1=c(1,1,0,0), indiv2=c(NA,1,1,0), indiv3=c(NA,0,3, NA))
> x <- new("genlight", dat)
> x
        /// GENLIGHT OBJECT /////////
         // 3 genotypes,  4 binary SNPs, size: 3.6 Kb
```

```
3 (25 %) missing data
// Basic content
  @gen: list of 3 SNPbin
// Optional content
  @ind.names:  3 individual labels
  @other: a list containing: elements without names
```

همانطور که مشاهده می‌شود شیء genlight ایجاد شده، متشکل از سه ژنوتیپ است و هر ژنوتیپ از چهار SNP دودوئی تشکیل شده است که مربوط به چهار لوکوس می‌باشند.

با استفاده از تابع names(x) می‌توان محتویات شیء genlight را مشاهده نمود و سپس با استفاده از $ هر یک از اجزاء لیست را فراخوانی نمود:

```
> names(x)
 [1] "gen"        "n.loc"      "ind.names"  "loc.names"  "loc.all"
 [6] "chromosome" "position"   "ploidy"     "pop"        "strata"
[11] "hierarchy"  "other"
```

دستور x$n.loc، تعداد لوکوس‌ها را نشان می‌دهد که معادل تابع nLoc(x) می‌باشد:

```
> x$n.loc
[1] 4
```

دستور x$ind.names ، اسم لوکوس‌ها را نشان می‌دهد که معادل تابع indNames(x) می‌باشد:

```
> x$ind.names
[1] "indiv1" " indiv2" " indiv3"
```

با استفاده از دستور locNames، می‌توان اسامی لوکوس‌ها را مشاهده یا آنها را نام‌گذاری کرد:

```
> locNames(x) <- paste("SNP",1:nLoc(x),sep=".")
> locNames(x)
[1] "SNP.1" "SNP.2" "SNP.3" "SNP.4"
```

با استفاده از توابع as.list و as.matrix می‌توان شیء genlight را به ترتیب به لیست و ماتریس تبدیل کرد:

```
> as.list(x)
        $indiv1
        [1] 1 1 0 0

        $indiv2
        [1] NA  1  1  0

        $indiv3
        [1] NA  0  3 NA
> as.matrix(x)
        SNP.1   SNP.2   SNP.3   SNP.4
indiv1      1       1       0       0
indiv2     NA       1       1       0
indiv3     NA       0       3      NA
```

برای سطح پلوئیدی چنانچه تعیین نشده باشد، بزرگترین عدد برای هر فرد در بین لوکوس‌های SNP مربوط به آن فرد (بزرگترین عدد در هر ردیف)، به عنوان سطح پلوئیدی در نظر گرفته می‌شود. برای تعریف پلوئیدی می‌توان از تابع ()ploidy استفاده نمود:

```
> ploidy(x)
 indiv1  indiv2  indiv3
      1       1       3
> ploidy(x)<-4
> ploidy(x)
 indiv1  indiv2  indiv3
      4       4       4
```

با استفاده از علامت کروشه [] می‌توان زیرمجموعه‌ای از داده‌های genlight ایجاد کرد. به عنوان مثال برای جدا کردن فرد اول و سوم، می‌توان دستور زیر را اجرا کرد:

```
> x[c(1,3)]
        /// GENLIGHT OBJECT ////////
```

```
// 2 genotypes,  4 binary SNPs, size: 3 Kb
2 (25 %) missing data
// Basic content
  @gen: list of 2 SNPbin
  @ploidy: ploidy of each individual  (range: 4-4)
// Optional content
  @ind.names:  2 individual labels
  @loc.names:  4 locus labels
  @other: a list containing: elements without names
> as.matrix(x[c(1,3)])
       SNP.1 SNP.2 SNP.3 SNP.4
indiv1     1     1     0     0
indiv3    NA     0     3    NA
```

برای تمرین با داده‌های SNP با ابعادی نزدیک به واقع، به مثال زیر توجه کنید. بدین منظور تعداد یک میلیون جایگاه SNP برای ۵۰ فرد (ژنوتیپ)، بطور تصادفی تعریف می‌کنیم. برای این کار لازم است برای هر فرد به تعداد یک میلیون بار (1e6) بطور تصادفی و با جای‌گذاری، از وکتور (0,1,NA) که مربوط به شمارش تعداد آلل دوم در هر جایگاه SNP است، نمونه‌برداری انجام شود. همچنین از آنجا که فراوانی وقوع سه عنصر وکتور (0,1,NA) در جایگاه‌های SNP یکسان نیست، می‌توان برای انجام نمونه‌برداری از سه عنصر وکتور مذکور، مقدار احتمال (بطور مثال، 0.01 ,0.49 ,0.5) در نظر گرفت. و سپس این عمل را بایستی برای ۵۰ فرد تکرار کنیم. بنابراین تابعی به شکل زیر تعریف می‌کنیم:

```
function(i) sample(c(0,1,NA), 1e6, prob=c(.5, .49, .01), replace=TRUE)
```

و آنرا برای ۵۰ فرد اعمال می‌کنیم (ممکن است کمی زمان‌بر باشد):

```
> dat <- lapply(1:50, function(i) sample(c(0,1,NA), 1e6, prob=c(.5, .49, .01), replace=TRUE))
> class(dat)
    [1] "list"
> length(dat)
    [1] 50
```

همانطور که مشاهده می‌شود شیء ایجاد شده (dat) از نوع لیست و متشکل از ۵۰ جزء است که هر جزء اطلاعات مربوط به یک ژنوتیپ را در بر دارد. می‌توانیم ژنوتیپ‌ها را نام‌گذاری کنیم:

```
> names(dat) <- paste("indiv", 1:length(dat))
> names(dat)
 [1] "indiv 1"  "indiv 2"  "indiv 3"  "indiv 4"  "indiv 5"  "indiv 6"
 [7] "indiv 7"  "indiv 8"  "indiv 9"  "indiv 10" "indiv 11" "indiv 12"
[13] "indiv 13" "indiv 14" "indiv 15" "indiv 16" "indiv 17" "indiv 18"
[19] "indiv 19" "indiv 20" "indiv 21" "indiv 22" "indiv 23" "indiv 24"
[25] "indiv 25" "indiv 26" "indiv 27" "indiv 28" "indiv 29" "indiv 30"
[31] "indiv 31" "indiv 32" "indiv 33" "indiv 34" "indiv 35" "indiv 36"
[37] "indiv 37" "indiv 38" "indiv 39" "indiv 40" "indiv 41" "indiv 42"
[43] "indiv 43" "indiv 44" "indiv 45" "indiv 46" "indiv 47" "indiv 48"
[49] "indiv 49" "indiv 50"
```

اکنون می‌توان لیست dat را به شیء genlight تبدیل کرد (ممکن است کمی زمان‌بر باشد):

```
> x <- new("genlight", dat)
> x
 /// GENLIGHT OBJECT /////////
 // 50 genotypes,  1,000,000 binary SNPs, size: 7.9 Mb
 499009 (1 %) missing data
 // Basic content
   @gen: list of 50 SNPbin
 // Optional content
   @ind.names:  50 individual labels
   @other: a list containing: elements without names
```

با مقایسه حجم اشیاء dat و x، مربوط به قبل و بعد از تبدیل به genlight، مشاهده می‌شود که عمل تبدیل به genlight از کارایی بالایی در کاهش حجم داده‌ها (حدود ۵۰ برابر کاهش) برخوردار بوده است:

```
> print(object.size(dat), unit="aut")
```

```
381.5 Mb
> print(object.size(x), unit="au")
7.9 Mb
> object.size(dat)/object.size(x)
48.2739599590447 bytes
```

می‌توان تعدادی جمعیت تعریف کرد و افراد را به آن‌ها منتسب نمود. به عنوان مثال سه جمعیت Pop1، Pop2 و Pop3 بطور تصادفی برای ۵۰ فرد تعیین می‌شود:

```
> px<-sample(c("Pop1","Pop2","Pop3"), 50, replace=TRUE)
> px
 [1] "Pop3" "Pop1" "Pop1" "Pop1" "Pop2" "Pop2" "Pop3" "Pop1" "Pop1" "Pop2"
[11] "Pop3" "Pop1" "Pop2" "Pop1" "Pop2" "Pop3" "Pop1" "Pop2" "Pop2" "Pop2"
[21] "Pop1" "Pop2" "Pop2" "Pop2" "Pop3" "Pop2" "Pop1" "Pop2" "Pop3" "Pop3"
[31] "Pop3" "Pop2" "Pop3" "Pop2" "Pop3" "Pop3" "Pop1" "Pop3" "Pop1" "Pop2"
[41] "Pop2" "Pop2" "Pop1" "Pop1" "Pop3" "Pop2" "Pop1" "Pop3" "Pop2" "Pop2"
```

توجه داشته باشید از آنجا که تابع sample تصادفی عمل می‌نماید، نتایج شما ممکن است با نتایج بالا منطبق نباشد.

می‌توان از تابع pop برای افزودن جمعیت‌های فوق به شیء x، استفاده نمود:

```
> pop(x)<- px
> pop(x)
 [1] Pop3 Pop1 Pop1 Pop1 Pop2 Pop2 Pop3 Pop1 Pop1 Pop2 Pop3 Pop1 Pop2 Pop1 Pop2
[16] Pop3 Pop1 Pop2 Pop2 Pop2 Pop1 Pop2 Pop2 Pop2 Pop3 Pop2 Pop1 Pop2 Pop3 Pop3
[31] Pop3 Pop2 Pop3 Pop2 Pop3 Pop3 Pop1 Pop3 Pop1 Pop2 Pop2 Pop2 Pop1 Pop1 Pop3
[46] Pop2 Pop1 Pop3 Pop2 Pop2
Levels: Pop1 Pop2 Pop3
> x
/// GENLIGHT OBJECT /////////
// 50 genotypes,  1,000,000 binary SNPs, size: 7.9 Mb
499009 (1 %) missing data
```

```
// Basic content
  @gen: list of 50 SNPbin
// Optional content
  @ind.names:  50 individual labels
  @pop: population of each individual (group size range: 14-21)
  @other: a list containing: elements without names
```

همانطور که در نتایج بالا مشخص است، جمعیت‌های تعیین شده، به شیء x اضافه شده است (@pop) که اندازه آنها از ۱۴ تا ۲۱ فرد متغیر می‌باشد.

با استفاده از تابع seppop(x) می‌توان افراد جمعیت‌ها را از یکدیگر متمایز نمود. دستورات زیر را اجرا، و نتایج را مشاهده و مقایسه نمایید:

```
> seppop(x)
> seppop(x)$Pop1
> seppop(x)$Pop2
> seppop(x)$Pop3
```

توابع تکمیلی برای اشیاء genlight

با استفاده از توابع تکمیلی، انجام برخی محاسبات مربوط به اشیاء genlight امکان‌پذیر است. مهمترین این توابع عبارتند از:

glSum: حاصلِ‌جمع تعداد آلل دوم را در هر جایگاه SNP محاسبه می‌کند.

glNA: تعداد مقادیر گمشده در هر جایگاه SNP محاسبه می‌کند.

glMean: میانگین تعداد آلل دوم را در هر جایگاه SNP محاسبه می‌کند.

glVar: واریانس تعداد آلل دوم را در هر جایگاه SNP محاسبه می‌کند.

به عنوان مثال، شش جایگاه SNP را در سه ژنوتیپ، به صورت زیر تعریف می‌کنیم:

```
> x <- new("genlight", list(c(0,0,1,1,0), c(1,1,1,0,0,1), c(2,1,1,1,1,NA)))
  Warning message:
  In .local(.Object, ...) :
```

```
Genotypes have variable length; completing shorter ones with NAs.
```

دلیل پیام اخطار، طول متفاوت وکتور اول از دو وکتور دیگر است (وکتور اول دارای پنج عضو و دو وکتور دیگر، دارای شش عضو می‌باشند). کمبود عضو مذکور در وکتور اول، به عنوان مقدار گمشده (NA) در نظر گرفته می‌شود.

```
> x
```

```
/// GENLIGHT OBJECT /////////
// 3 genotypes,  6 binary SNPs, size: 3.5 Kb
2 (11.11 %) missing data
// Basic content
  @gen: list of 3 SNPbin
// Optional content
  @other: a list containing: elements without names
```

```
> as.matrix(x)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    1    1    0   NA
[2,]    1    1    1    0    0    1
[3,]    2    1    1    1    1   NA
```

به عدد ۲ در جایگاه SNP اول در ژنوتیپ سوم، توجه داشته باشید. این عدد نشان می‌دهد که ژنوتیپ سوم، (حداقل) دیپلوئید است. همانطور که اشاره شد سطح پلوئیدی افراد را می‌توان با استفاده از تابع ploidy مشخص نمود:

```
> ploidy(x)
[1] 1 1 2
> glNA(x)
[1] 0 0 0 0 0 3
> glSum(x)
[1] 3 2 3 2 1 1
> glMean(x)
[1] 0.75 0.50 0.75 0.50 0.25 1.00
```

توجه داشته باشید که محاسبات فوق، از جمله تعداد داده‌های گمشده در جایگاه SNP ششم (۳)، با فرض دیپلوئید بودن ژنوتیپ سوم، محاسبه شده است.

جداکردن بخشی از لوکوس‌های SNP

برای جداکردن اطلاعات لوکوس‌ها از یکدیگر در اشیاء genlight نیز می‌توان از تابع seploc استفاده نمود. در این حالت بلوک‌هایی از SNPها تشکیل می‌شود. با استفاده از آرگومان‌های n.block و block.size می‌توان تعداد این بلوک‌ها و یا اندازه آنها را تعیین نمود.

به عنوان مثال، ابتدا توسط تابع glSim، ۱۰۰ فرد دیپلوئید با ۱۰۰۰ لوکوس SNP را شبیه‌سازی می‌نماییم (در مورد تابع glSim و نحوه کار با آن در بخش‌های بعد توضیح داده شده است):

```
> x <- glSim(100, 1000, 0, ploidy=2)
> class(x)
        [1] "genlight"
        attr(,"package")
        [1] "adegenet"
> nInd(x)
        [1] 100
> nLoc(x)
        [1] 1000
```

با استفاده از تابع glSum، فراوانی دومین آلل را در هر لوکوس محاسبه می‌کنیم:

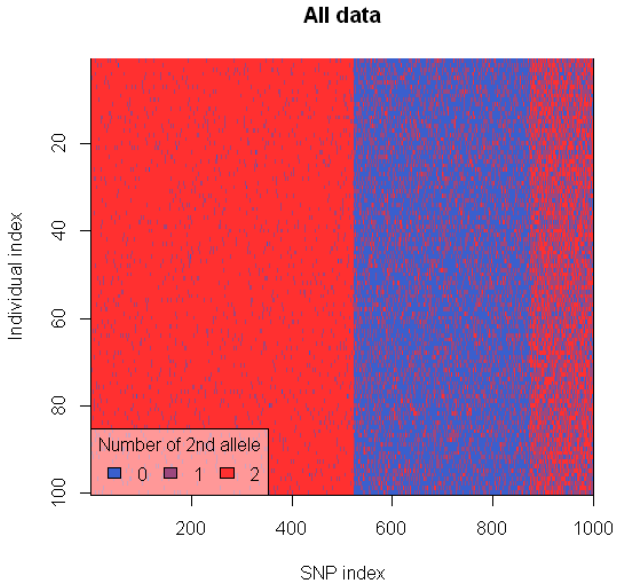
```
> head(glSum(x))
        [1] 195 196 196 192 197 196
```

با استفاده از تابع order، لوکوس‌ها را براساس فراوانی دومین آلل (از فراوانی بزرگ به کوچک) مرتب می‌نماییم.

```
> x <- x[,order(glSum(x))]
```

با استفاده از تابع glPlot، نمودار فراوانی دومین آلل را در افراد (شکل ۱۱-۱) ترسیم می‌نماییم (در مورد تابع glplot و نحوه کار با آن، در بخش‌های بعد توضیح داده شده است):

```
> glPlot(x, main="All data")
```



شکل ۱۱-۱ – نمودار فراوانی دومین آلل در ۱۰۰۰ لوکوس SNP شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

با استفاده از تابع seploc و آرگومان n.block=3، سه بلوک مجزا از مجموعه داده‌های SNP تشکیل می‌دهیم (توجه کنید که آرگومان parallel، تعیین می‌کند آیا از چندین هسته برای ایجاد داده‌های شبیه‌سازی شده استفاده شود (TRUE) یا خیر (FALSE). این گزینه مقدار زمان لازم برای محاسبات شبیه‌سازی داده‌ها را کاهش می‌دهد، ولی بر روی سیستم Windows پشتیبانی نمی‌شود):

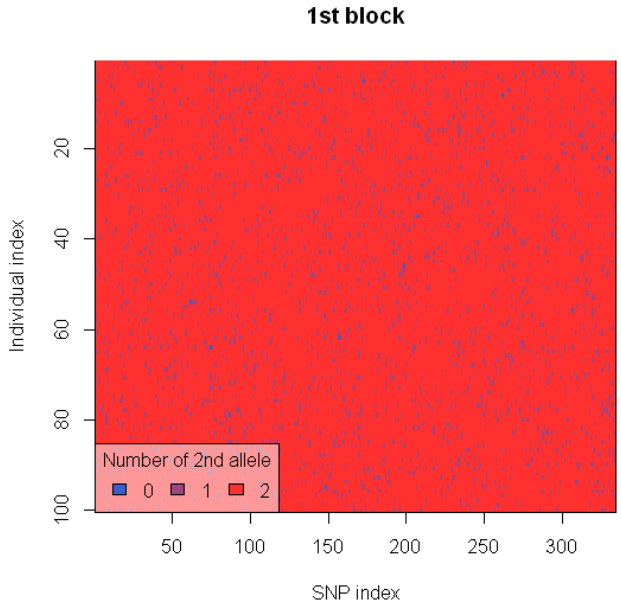
```
> foo <- seploc(x, n.block=3, parallel = FALSE)
> class(foo)
        [1] "list"
> foo
        $block.1
         /// GENLIGHT OBJECT /////////

         // 100 genotypes,  334 binary SNPs, size: 85.6 Kb
         0 (0 %) missing data
         // Basic content
           @gen: list of 100 SNPbin
           @ploidy: ploidy of each individual  (range: 2-2)
         // Optional content
           @other: a list containing: ancestral.pops


        $block.2
         /// GENLIGHT OBJECT /////////
         // 100 genotypes,  333 binary SNPs, size: 85.6 Kb
         0 (0 %) missing data
         // Basic content
           @gen: list of 100 SNPbin
           @ploidy: ploidy of each individual  (range: 2-2)
         // Optional content
```

```
  @other: a list containing: ancestral.pops


$block.3
 /// GENLIGHT OBJECT /////////
 // 100 genotypes,  333 binary SNPs, size: 85.6 Kb
 0 (0 %) missing data

 // Basic content
   @gen: list of 100 SNPbin
   @ploidy: ploidy of each individual  (range: 2-2)
 // Optional content
   @other: a list containing: ancestral.pops
```

همانطور که مشاهده می‌شود لیستی (foo) متشکل از سه جزء تشکیل می‌شود، که هر جزء خود یک شیء از نوع genlight است و تعداد لوکوس‌های SNP در این سه شیء genlight، تقریباً برابر (به ترتیب ۳۳۴، ۳۳۳ و ۳۳۳ لوکوس) است.

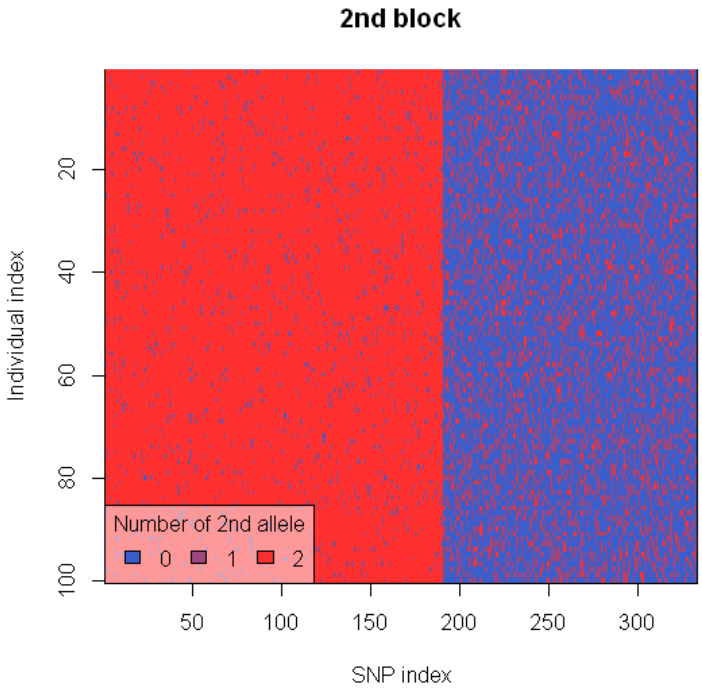
می‌توان با استفاده از تابع glPlot، نمودار فراوانی دومین آلل را در هریک از این بلوک‌ها ترسیم نمود (شکل‌های ۱۱-۲، ۱۱-۳ و ۱۱-۴):

```
> glPlot(foo[[1]], main="1st block")
```



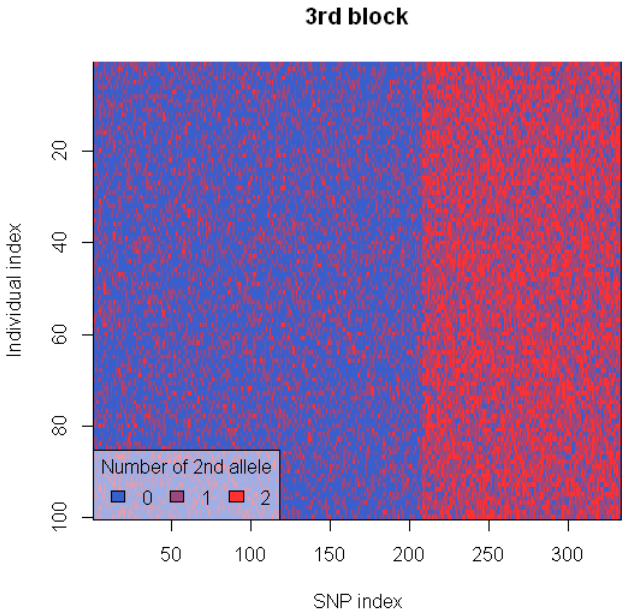
شکل ۱۱-۲ – نمودار فراوانی دومین آلل در بلوک اول (۳۳۴تایی) از ۱۰۰۰ لوکوس SNP شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

```
> glPlot(foo[[2]], main="2nd block")
```



شکل ۱۱-۳ – نمودار فراوانی دومین آلل در بلوک دوم (۳۳۳ تایی) از ۱۰۰۰ لوکوس SNP شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

```
> glPlot(foo[[3]], main="3rd block")
```



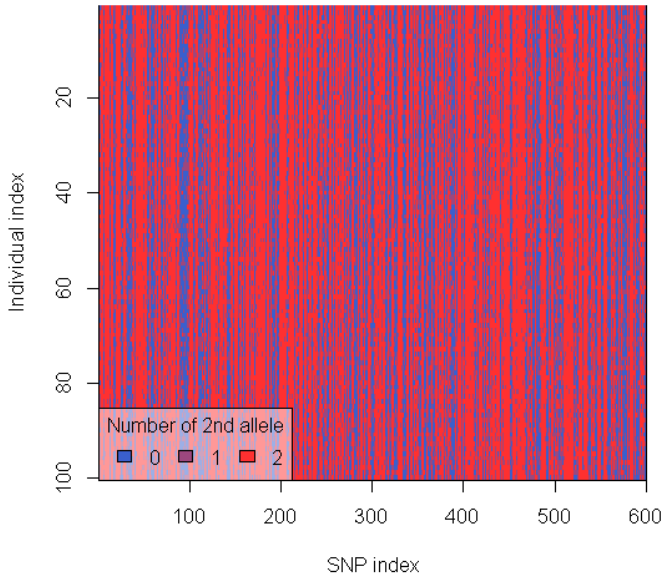
شکل ۱۱-۴ – نمودار فراوانی دومین آلل در بلوک سوم (۳۳۳تایی) از ۱۰۰۰ لوکوس SNP شبیه‌سازی در ۱۰۰ فرد دیپلوئید

بـا اسـتفاده از تـابع seploc و آرگومـان block.size=600، بلـوکی متشـکل از ۶۰۰ لوکـوس SNP تشـکیل می‌دهیم (شکل ۱۱-۵). آرگومان random=TRUE، تعیین می‌کنـد کـه انتخـاب لوکـوس‌هـای SNP بطـور تصادفی صورت پذیرد.

```
> foo <- seploc(x, block.size=600, random=TRUE, parallel = FALSE)
> foo
        $block.1
         /// GENLIGHT OBJECT /////////

         // 100 genotypes,  600 binary SNPs, size: 101.3 Kb
         0 (0 %) missing data

         // Basic content
           @gen: list of 100 SNPbin
           @ploidy: ploidy of each individual  (range: 2-2)

         // Optional content
           @other: a list containing: ancestral.pops


        $block.2
         /// GENLIGHT OBJECT /////////

         // 100 genotypes,  400 binary SNPs, size: 88.8 Kb
         0 (0 %) missing data

         // Basic content
           @gen: list of 100 SNPbin
           @ploidy: ploidy of each individual  (range: 2-2)

         // Optional content
           @other: a list containing: ancestral.pops
```

همانطور که مشاهده می‌شود لیست حاصل (foo) متشکل از دو شیء genlight است که شیء اول دارای ۶۰۰ لوکوس SNP است و شیء دوم باقیمانده لوکوس‌ها را (لوکوس ۴۰۰=۶۰۰-۱۰۰۰) در بر دارد.

```
> glPlot(foo[[1]])
```



شکل ۱۱-۵ – نمودار فراوانی دومین آلل در بلوک ۶۰۰تایی انتخاب شده به صورت تصادفی از ۱۰۰۰ لوکوس SNP شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

توجه داشته باشید دلیل عدم مشاهده الگوی پراکنش خاص در نمودار فوق (برخلاف نمودارهای قبلی)، استفاده از آرگومان random=TRUE، که همانطور که توضیح داده شد سبب می‌شود که انتخاب لوکوس‌های SNP بطور تصادفی صورت پذیرد. این آرگومان بصورت پیش‌فرض random=FALSE است که در اینحالت، لوکوس‌های SNP بطور تصادفی انتخاب نمی‌شوند، بلکه پیوستگی و ترتیب لوکوس‌ها (براساس ترتیب مندرج در شیء genlight مربوطه)، در هنگام جدا کردن لوکوس‌ها و قرار دادن آنها در اشیاء متمایز genlight (توسط تابع seploc) در نظر گرفته می‌شود.

با استفاده از تابع seppop می‌توان اشیاء genlight را نیز براساس جمعیت‌ها تفکیک نمود. به عنوان مثال یک شیء genlight با سه ژنوتیپ و ۱۰۰۰ لوکوس SNP تشکیل می‌دهیم و دو جمعیت pop1 و pop2 را به آنها منتسب می‌کنیم و سپس با استفاده از تابع seppop ژنوتیپ‌ها را برحسب جمعیت‌های مشترک، متمایز می‌نماییم:

```
> x <- new("genlight", list(a=rep(1,1e3),b=rep(0,1e3),c=rep(1, 1e3)))
> x
        /// GENLIGHT OBJECT /////////

        // 3 genotypes,  1,000 binary SNPs, size: 3.9 Kb
        0 (0 %) missing data

        // Basic content
```

```
  @gen: list of 3 SNPbin

 // Optional content
  @ind.names:  3 individual labels
  @other: a list containing: elements without names
```

```
> pop(x)
        NULL
> pop(x) <- c("pop1","pop2", "pop1")
> pop(x)
        [1] pop1 pop2 pop1
        Levels: pop1 pop2
> seppop(x)
        $pop1
         /// GENLIGHT OBJECT /////////
         // 2 genotypes,  1,000 binary SNPs, size: 3.2 Kb
         0 (0 %) missing data
         // Basic content
           @gen: list of 2 SNPbin
         // Optional content
           @ind.names:  2 individual labels
           @pop: population of each individual (group size range: 2-2)
           @other: a list containing: elements without names


        $pop2
         /// GENLIGHT OBJECT /////////
         // 1 genotypes,  1,000 binary SNPs, size: 2.4 Kb
         0 (0 %) missing data
         // Basic content
           @gen: list of 1 SNPbin
         // Optional content
           @ind.names:  1 individual labels
           @pop: population of each individual (group size range: 1-1)
           @other: a list containing: elements without names
```

```
> as.matrix(seppop(x)$pop1)[,1:20]
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]
a	1	1	1	1	1	1	1	1	1	1	1	1	1	1
c	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]
a	1	1	1	1	1	1
c	1	1	1	1	1	1

```
> as.matrix(seppop(x)$pop2)[,1:20,drop=FALSE]
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]
b	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	[,15]	[,16]	[,17]	[,18]	[,19]	[,20]
b	0	0	0	0	0	0

شبیه سازی داده‌های SNP

تابع glSim، شبیه‌سازی داده‌های SNP را با امکان ایجاد ساختارهای متباین بین دو گروه و همچنین امکان لحاظ کردن ساختارهای جمعیتی اجدادیِ پیش‌زمینه، میسر می‌سازد. شیء حاصل از این تابع از نوع genlight است:

glSim(n.ind, n.snp.nonstruc, n.snp.struc = 0, grp.size = c(0.5, 0.5), k = NULL, pop.freq = NULL, ploidy = 1, alpha = 0, parallel = FALSE, LD = TRUE)

n.ind: عددی که نشان دهندهِ تعداد افرادی است که باید شبیه‌سازی شوند.

n.snp.nonstruc: عددی که نشان دهندهِ تعداد SNPهای غیرساختاری است که باید شبیه‌سازی شوند. برای این نوع از SNPها، تمام افراد از توزیع دوجمله‌ای یکسانی استخراج می‌شوند.

n.snp.struc: عددی که نشان دهندهِ تعداد SNPهای ساختاری است که باید شبیه‌سازی شوند. برای این نوع از SNPها، توزیع‌های دوجمله‌ای متفاوتی برای دو گروهِ (پیش‌فرض) شبیه‌سازی شده، استفاده می‌شود و فراوانی‌های آلل‌ها در دو گروه، به نحوی تعیین می‌شود که سبب ایجاد تفاوت بین دو گروه شود.

grp.size: وکتوری حاوی دو عدد اعشاری که نشان دهندهِ نسبتِ اندازهِ دو گروه فنوتیپی است (در نتیجه جمع دو عدد باید برابر با یک باشد). پیش فرض مقدار این دو عدد، ۰/۵ در نظر گرفته شده است که بیانگر اندازه جمعیت مساوی است.

k: عددی که نشان دهندهِ تعداد جمعیت اجدادی است که باید ایجاد شود.

pop.freq: وکتوری با طول k که نشان دهندهِ نسبت اندازه‌ی k جمعیتِ اجدادی است (جمع دو عدد باید برابر با یک باشد). اگر بطور پیش‌فرض pop.freq=NULL، باشد ولی k، NULL نباشد، این نسبت بطور تصادفی ایجاد خواهد شد.

ploidy: عددی که نشان دهندهِ سطح پلوئیدیِ ژنوتیپ‌های شبیه‌سازی شده، است.

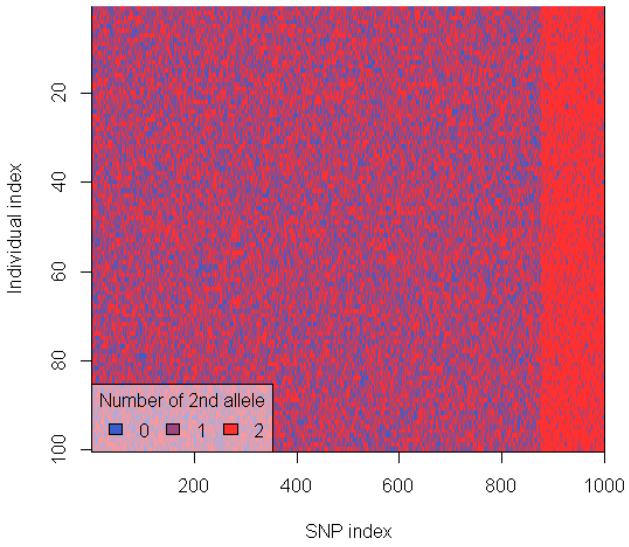
alpha: پارامتر عدم تقارن است و عبارت از مقدار عددی بین صفر و ۰/۵ می‌باشد که سبب الزام به ایجاد تفاوت آللی بین گروه‌ها می‌شود. قوی‌ترین تفاوت بین گروه‌ها در مقدار ۰/۵ از این پارامتر، و ضعیف‌ترین تفاوت بین گروه‌ها در مقدار صفر (پیش‌فرض) برقرار است.

parallel: یک گزاره منطقی که تعیین می‌کند آیا از چندین هسته برای ایجاد داده‌های شبیه‌سازی شده استفاده شود یا خیر. این گزینه مقدار زمان لازم برای محاسبات شبیه‌سازی داده‌ها را کاهش می‌دهد، ولی برروی سیستم Windows پشتیبانی نمی‌شود.

LD: یک گزاره منطقی که تعیین می‌کند آیا لوکوس‌ها، عدم تعادل لینکاژی نشان دهند (TRUE)، و یا اینکه بصورت مستقل ایجاد شوند (FALSE، پیش‌فرض). در صورت انتخاب TRUE، داده‌ها بصورت بلوک‌هایی از SNPهای همبسته، ایجاد می‌شوند.

مثال: شبیه‌سازی تعداد هزار SNP غیرساختاری در ۱۰۰ فرد دیپلوئید (شکل ۱۱-۶):

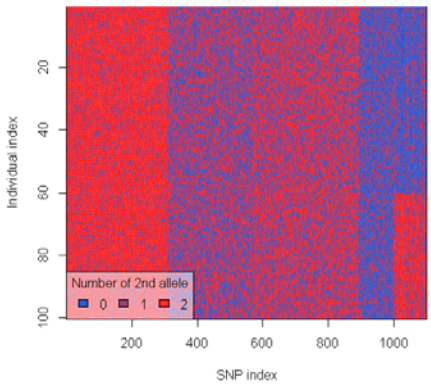
```
> x <- glSim(100, 1e3, ploid=2)
> plot(x)
```



شکل ۱۱-۶ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

مثال: شبیه‌سازی تعداد هزار SNP غیرساختاری و صد SNP ساختاری در ۱۰۰ فرد دیپلوئید (شکل ۱۱-۷):

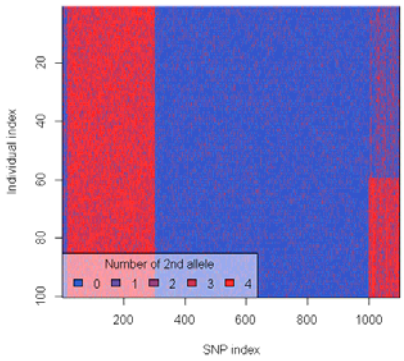
```
> x <- glSim(100, 1e3, n.snp.struc=100, ploid=2)
> plot(x)
```



شکل ۱۱-۷ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

مثال: شبیه‌سازی تعداد هزار SNP غیرساختاری و صد SNP ساختاری در ۱۰۰ فرد تتراپلوئید (شکل ۱۱-۸):

```
> x <- glSim(100, 1e3, n.snp.struc=100, ploid=4)
> plot(x)
```

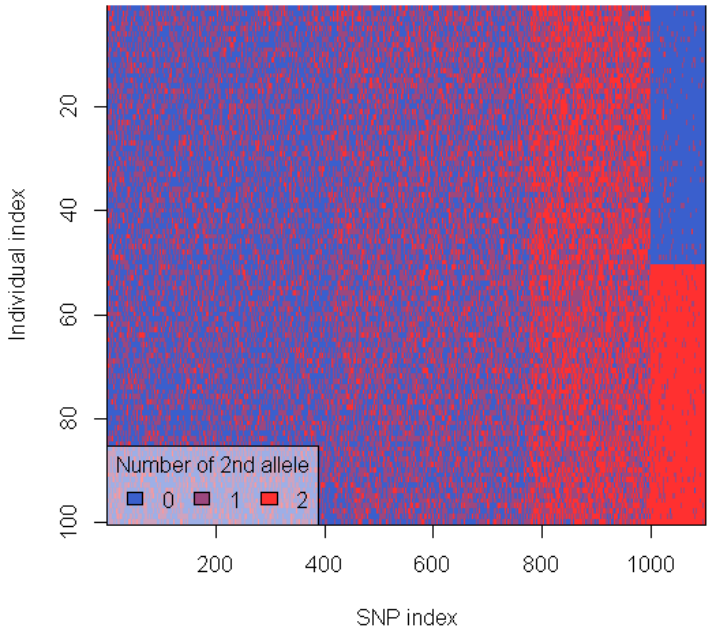


شکل ۱۱-۸ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد تتراپلوئید

مثال: شبیه‌سازی تعداد هزار SNP غیرساختاری و صد SNP ساختاری در ۱۰۰ فرد دیپلوئید با تفاوت‌های بین گروهی نسبتاً بزرگ (α=۰/۴، شکل ۱۱-۹):

```
> x <- glSim(100, 1e3, n.snp.struc=100, ploid=2, alpha=0.4)
```

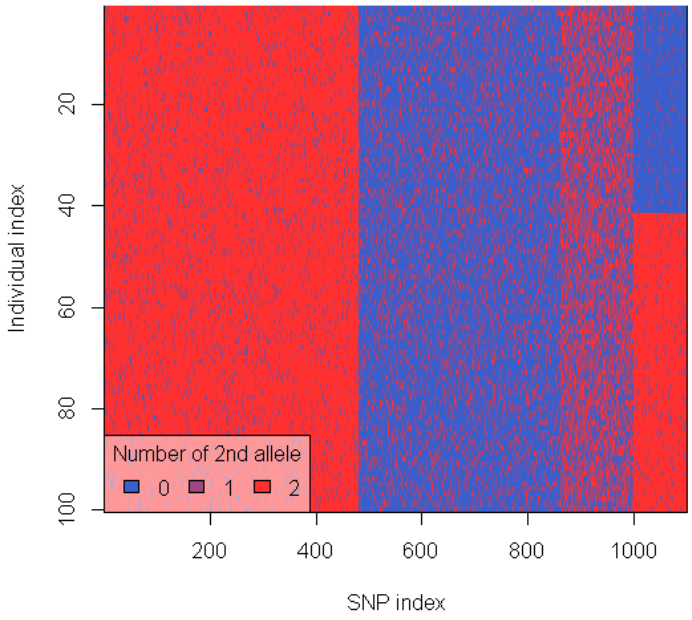
```
> plot(x)
```



شکل ۱۱-۹ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید با لحاظ کردن تفاوت بین گروهی نسبتاً بزرگ (α=۰/۴)

مثال: شبیه‌سازی تعداد هزار SNP غیرساختاری و صد SNP ساختاری در ۱۰۰ فرد دیپلوئید با تفاوت‌های بین گروهی نسبتاً بزرگ و قید ساختارهای عدم تعادل لینکاژی (شکل ۱۱-۱۰):

```
> x <- glSim(100, 1e3, n.snp.struc=100, ploid=2, alpha=0.4, LD=TRUE)
> plot(x)
```



شکل ۱۱-۱۰ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید با لحاظ کردن تفاوت بین گروهی نسبتاً بزرگ و ساختارهای عدم تعادل لینکاژی

رسم نمودار فراوانی داده‌های SNP

همانطور که در مثال‌های بخش‌های پیشین نشان داده شد به منظور نمایش داده‌های SNP در اشیاء genlight، می‌توان از تابع glPlot استفاده نمود. در اینجا به توضیح بیشتر در خصوص آرگومان‌های این تابع می‌پردازیم:

glPlot(x, col=NULL, legend=TRUE, posi="bottomleft", bg=rgb(1,1,1,.5),...)

x: یک شیء genlight است.

col: یک وکتور اختیاری برای رنگ است. مقدار اول مربوط به آلل‌های صفر (0) است. آخرین مقدار مربوط به سطح پلوئیدی داده‌ها است. بنابراین وکتور مذکور باید طولی برابر با ploidy(x)+1 داشته باشد.

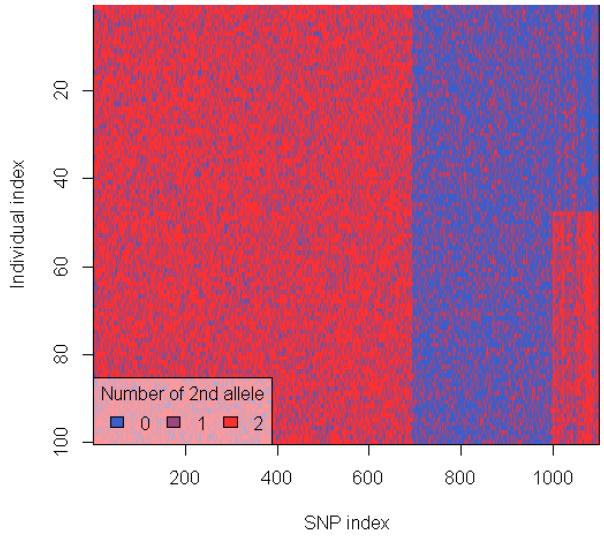
legend: گزاره‌ای منطقی است که مشخص می‌کند آیا برچسب شرح علائم به نمودار اضافه شود یا خیر.

posi: یک رشته کاراکتری است که مشخص می‌کند برچسب شرح علائم در کجای نمودار درج شود که با ترکیب یکی از هر یک از موارد (top ،bottom) و (right ، left)، در چهار ناحیه قابل درج می‌باشد.

bg: رنگی که به عنوان پیش‌زمینه در برچسب شرح علائم بکار می‌رود که بصورت پیش‌فرض، سفیدِ شفاف تعیین شده است که البته ممکن است در برخی از دستگاه‌ها پشتیبانی نشود و دراینصورت باید رنگ پیش‌زمینه مشخص شود (مثلاً بصورت: "bg="white)

به عنوان مثال، ابتدا با استفاده از تابع glSim، تعداد هزار SNP غیرساختاری و صد SNP ساختاری را در ۱۰۰ فرد دیپلوئید، شبیه‌سازی و سپس نمودار داده‌های SNP را ترسیم می‌نماییم (شکل ۱۱-۱۱):

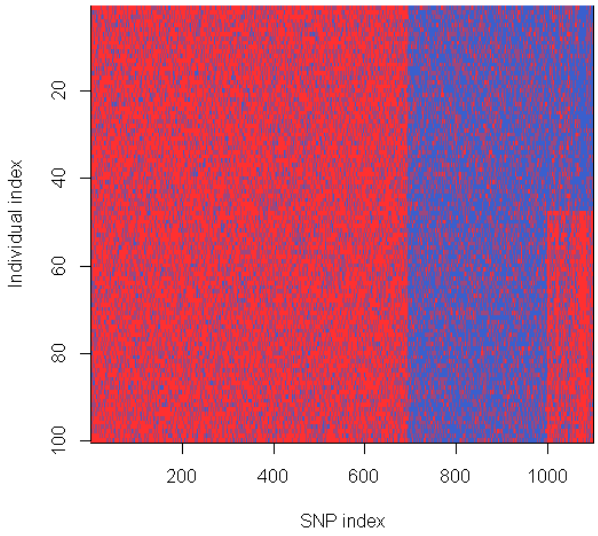
```
> x <- glSim(100, 1e3, n.snp.struc=100, ploid=2)
> glPlot(x)
```



شکل ۱۱-۱۱ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

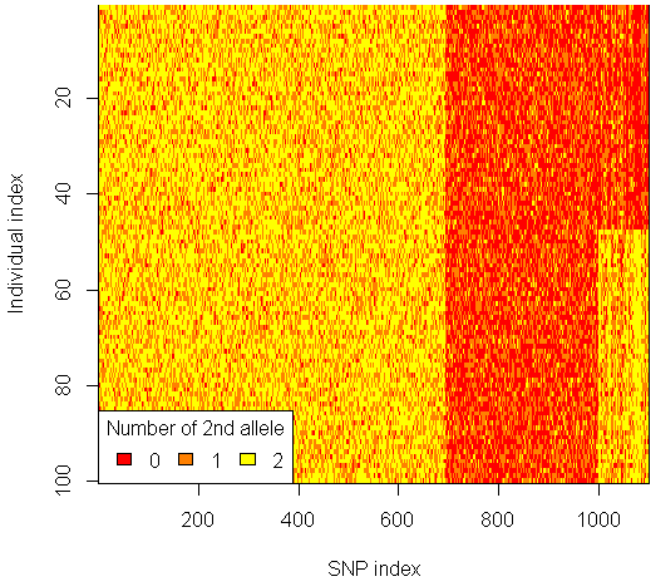
می‌توانیم برچسب شرح علائم را حذف و یا از رنگ‌های دیگری استفاده کنیم (شکل ۱۱-۱۲ و ۱۱-۱۳):

```
> glPlot(x, leg=FALSE)
```



شکل ۱۱-۱۲ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید با حذف برچسب شرح علائم

```
> glPlot(x, col=heat.colors(3), bg="white")
```



شکل ۱۱-۱۳ – نمودار فراوانی دومین آلل هزار SNP غیرساختاری و صد SNP ساختاری شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید با تغییر رنگ پیش‌فرض

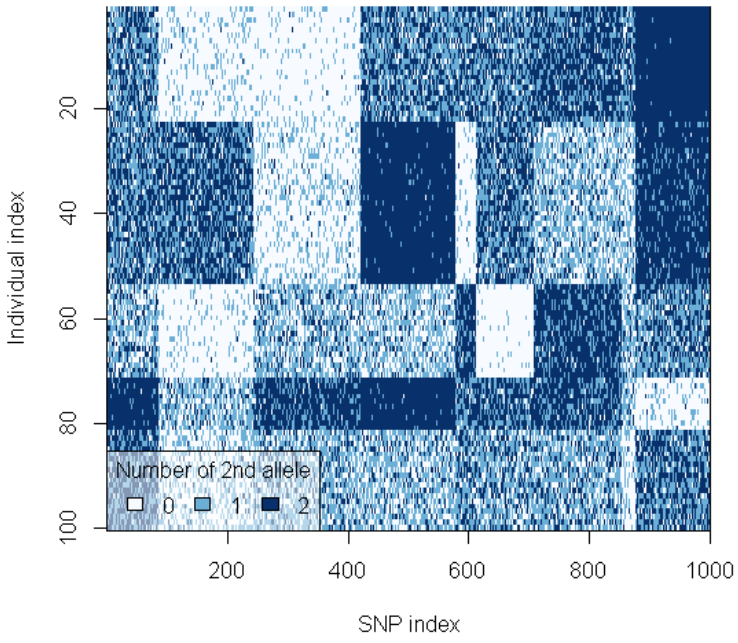
نکته: استفاده از تابع plot (بجای glPlot) نتایج مشابهی ایجاد خواهد نمود و لذا دو تابع plot و glPlot، در این موارد همتا می‌باشند.

یکی از کاربردهای نمودارِ تعداد آللِ جایگاه‌های SNP در افراد، به دست آوردن تصویری از ساختاریافتگی احتمالی جمعیت و همچنین عدم تعادل لینکاژی بین لوکوس‌های SNP است. برای نشان دادن این موضوع، با استفاده از شبیه‌سازی، جمعیتی ساختار یافته ایجاد می‌کنیم و نمودار تعداد آللِ جایگاه‌های SNP در افراد را رسم نماییم. به عنوان مثال با دستور زیر، تعداد هزار SNP غیرساختاری در پنج جمعیت اجدادی را در ۱۰۰ فرد دیپلوئید، شبیه‌سازی می‌نماییم (شکل ۱۱-۱۴):

```
> x <- glSim(100, 1000, k=5, block.maxsize=200, ploidy=2, sort.pop=TRUE)
```

توجه کنید که انتخاب گزینه TRUE برای آرگومان sort.pop سبب می‌شود که افراد براساس جمعیت اجدادی به ترتیب مرتب شوند.

```
> glPlot(x, col=bluepal(3))
```



شکل ۱۱-۱۴ – نمودار فراوانی دومین آلل در هزار SNP غیرساختاری با لحاظ کردن پنج جمعیت اجدادی شبیه‌سازی شده در ۱۰۰ فرد دیپلوئید

در نمودار فوق، بلوک‌هایی که بصورت ستونی ظاهر شده‌اند، نشانه‌ای از وجود ساختار ژنتیکی در جمعیت می‌باشد و بلوک‌هایی که بصورت افقی ظاهر شده‌اند، عدم تعادل لینکاژی در دامنه SNPهای مجاور را نشان می‌دهند. البته این مشاهدات صرفاً به عنوان مقدمه‌ای برای انجام تجزیه‌های جزئی‌تر در این زمینه خواهد بود.

خوانش فایل داده‌های SNP

پکیج adegenet برای ذخیره داده‌های دوآللی SNP در قالب فایل‌های متنی، دارای فرمتی اختصاصی با پسوند snp. می‌باشد. این فرمت دارای چندین مزیت است از جمله اینکه در مقایسه با فرمت‌های غیرفشرده‌ی معمول، تقریباً فشرده می‌باشد که در نتیجه، ذخیره سایر اطلاعات در مورد افراد یا لوکوس‌ها را میسر می‌سازد. مزیت مهم دیگر اینست که در این فرمت، نیازی نیست که همه اطلاعات به یکباره خوانش شود و این ویژگی سبب به حداقل رسیدن نیاز به RAM هنگام ورود داده‌ها می‌شود. برای آشنایی با این فرمت یک فایل نمونه به نام exampleSnpDat در پکیج adegenet قرار داده شده است که با استفاده از دستور زیر قابل مشاهده می‌باشد:

```
> file.show(system.file("files/exampleSnpDat.snp",package="adegenet"))
```

فایل‌های snp. با استفاده از تابع read.snp قابل خوانش می‌باشند. تابع read.snp، داده‌های فایل‌هایsnp. را به اشیاء genlight تبدیل می‌کند. این تابع، عمل خوانش را در چند مرحله انجام می‌دهد و در هر مرحله داده‌های بخشی از افراد (chunk) خوانش می‌شود. این طرز عمل باعث کاهش نیاز به RAM (در مقایسه با خوانش یکباره داده‌ها) می‌شود و البته میزان محاسبات را افزایش خواهد داد. آرگومان chunkSize، تعداد ژنومی را که در هربار بایستی خوانش شود را تعیین می‌کند. در نظر گرفتن مقادیر بالاتر برای این آرگومان سبب خوانش سریع‌تر می‌شود ولی درعین حال نیاز به RAM بیشتری نیز دارد. برای مشاهده عمل و خروجی تابع read.snp، دستورات زیر را اجرا و نتایج را مشاهده نمایید:

```
> obj <- read.snp(system.file("files/exampleSnpDat.snp",package="adegenet"), chunk=2, parallel=FALSE)

> as.matrix(obj, parallel=FALSE)
```

	1.a/t	8.g/c	11.a/c	43.t/a
foo	1	0	2	0
bar	0	0	1	2

```
toto                                            1  0   NA  0
Nyarlathotep                                    0  1   2   0
an even longer label but OK since on a single line 1  1   0   0
> alleles(obj)
[1] "a/t" "g/c" "a/c" "t/a"
> indNames(obj)
[1] "foo"
[2] "bar"
[3] "toto"
[4] "Nyarlathotep"
[5] "an even longer label but OK since on a single line"
```

داده‌های SNP در سطح ژنوم، اغلب توسط نرم‌افزار PLINK (Purcell *et al.*, 2007) خوانش می‌شوند که در نتیجه سبب تبدیل فرمت raw. به استاندارد رایج شده است. داده‌های با فرمت raw. را می‌توان با استفاده از تابع read.PLINK خوانش کرده و به اشیاء genlight تبدیل نمود.

همچنین داده‌هایی که بصورت همردیفی هستند با استفاده از تابع fasta2genlight قابل خوانش و تبدیل به اشیاء genlight می‌باشند. این تابع، SNPها را از فایل‌های fasta (با پسوند fasta.، fas. یا fa.) استخراج می‌کند. برای آشنایی با این فرمت یک فایل نمونه به نام usflu.fasta در پکیج adegenet قرار داده شده است. مسیر فایل را می‌توان با استفاده از تابع system.file بدست آورد و سپس تابع fasta2genlight را بر روی آن اعمال نمود (ممکن است قدری بطول بیانجامد):

```
> myPath <- system.file("files/usflu.fasta",package="adegenet")
> flu <- fasta2genlight(myPath, chunk=10, parallel=FALSE)
  Converting FASTA alignment into a genlight object...
  Looking for polymorphic positions...
  .............................
  Extracting SNPs from the alignment...
  .............................
  Building final object...
```

```
...done.
```

```
> flu
```

```
/// GENLIGHT OBJECT /////////

// 80 genotypes,  274 binary SNPs, size: 68.5 Kb
26 (0.12 %) missing data

// Basic content
  @gen: list of 80 SNPbin
  @ploidy: ploidy of each individual  (range: 1-1)

// Optional content
  @ind.names:  80 individual labels
  @loc.all:  274 alleles
  @position: integer storing positions of the SNPs
  @other: a list containing: elements without names
```

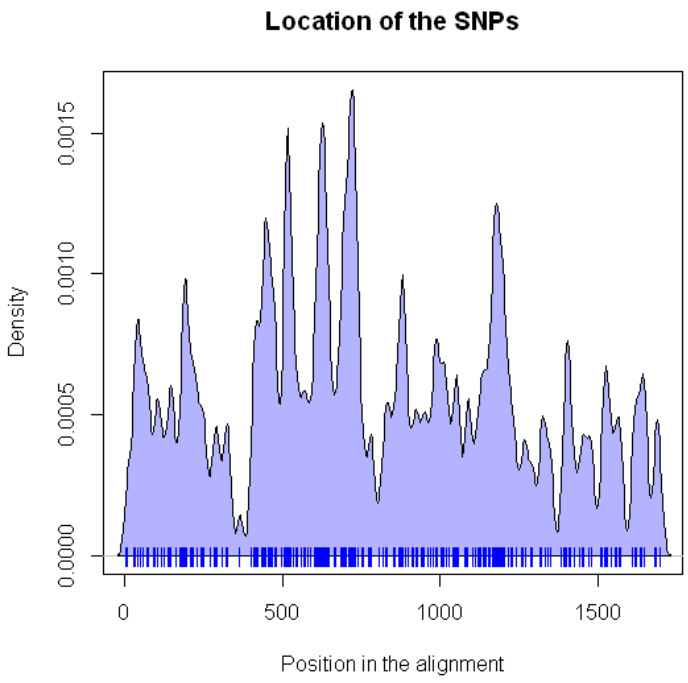
شیء genlight فوق شامل داده‌های SNP مربوط قطعه توالی هماگلوتنین (HA) در ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2) است که در طی دو دهه در ایالات متحده آمریکا نمونه برداری شده‌اند. همچنین شیء فوق حاوی اطلاعات مربوط به مکان قرار گرفتن SNPها و همچنین آلل‌های مربوط به هر لوکوس SNP است که به ترتیب با استفاده از توابع position و alleles قابل مشاهده می‌باشند. لوکوس‌های SNP نیز با ترکیبی از جایگاه قرار گرفتن آنها و آلل‌های مربوطه، نام‌گذاری می‌شوند که با تابع locNames قابل مشاهده می‌باشد:

```
> head(position(flu), 20)
  [1]   7  12  31  32  36  37  44  45  52  60  62  72  73  78  96  99 105 108 121
 [20] 128
> head(alleles(flu), 20)
```

```
 [1] "a/g" "c/t" "t/c" "t/c" "t/c" "c/a" "t/c" "c/t" "a/g" "c/t" "g/t" "c/a"
[13] "a/g" "a/g" "a/g" "c/t" "a/g" "g/a" "c/a" "a/g"
> head(locNames(flu), 20)
 [1] "7.a/g"   "12.c/t"  "31.t/c"  "32.t/c"  "36.t/c"  "37.c/a"  "44.t/c"
 [8] "45.c/t"  "52.a/g"  "60.c/t"  "62.g/t"  "72.c/a"  "73.a/g"  "78.a/g"
[15] "96.a/g"  "99.c/t"  "105.a/g" "108.g/a" "121.c/a" "128.a/g"
```

با ترکیبی از توابع density و position و تعریف یک دامنه (آرگومان bw) می‌توان توزیع تراکم SNPها را در طول ژنوم ترسیم نمود (شکل ۱۱-۱۵):

```
> temp <- density(position(flu), bw=10)
> plot(temp, type="n", xlab="Position in the alignment", main="Location of the SNPs", xlim=c(0,1701))
> polygon(c(temp$x,rev(temp$x)), c(temp$y, rep(0,length(temp$x))), col=transp("blue",.3))
> points(position(flu), rep(0, nLoc(flu)), pch="|", col="blue")
```



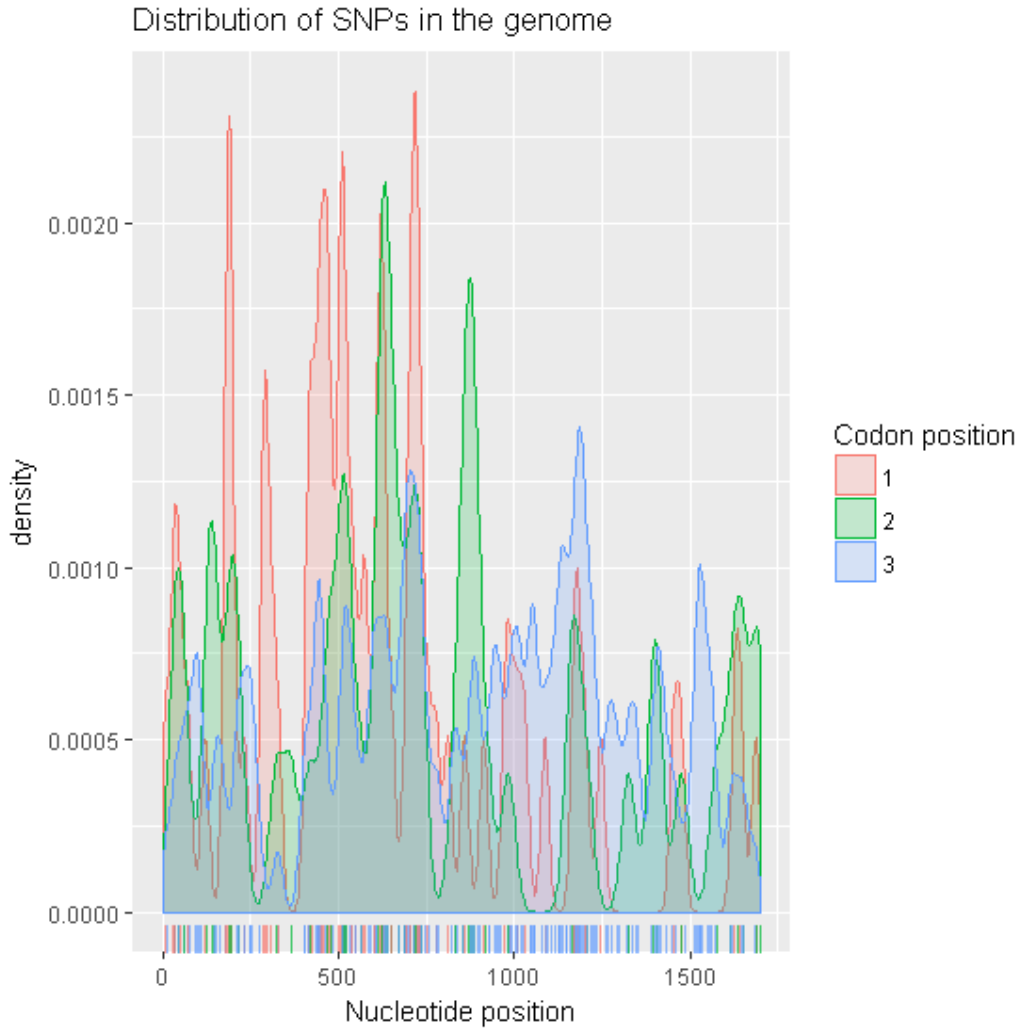
شکل ۱۱-۱۵ - توزیع تراکم SNPها در قطعه توالی هماگلوتنین (HA) از ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2)

توجه: استفاده از تابع snpposi.plot نمودار مشابهی ایجاد خواهد نمود:

```
> snpposi.plot(position(flu), genome.size=1700, codon=FALSE)
```

آرگومان codon بطور پیش فرض TRUE است. درصورت اجرای تابع snpposi.plot، با TRUE بودن این آرگومان، نمودار تراکم SNPها به تفکیک کدون ترسیم خواهد شد (شکل ۱۱-۱۶):

```
> snpposi.plot(position(flu), genome.size=1700, codon=TRUE)
```



شکل ۱۱-۱۶ - توزیع تراکم SNPها در قطعه توالی هماگلوتنین (HA) از ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2) به تفکیک کدون

با توجه به این نتایج به نظر می‌رسد که SNPها، به استثناء تعداد محدودی کانون تراکم، در فاصله‌ی نوکلئوتیدهای ۴۰۰ تا ۷۰۰، بطور تقریباً یکنواختی در طول قطعه HA توزیع شده‌اند. این موضوع را می‌توان با استفاده از تابع snpposi.test آزمون کرد:

```
> snpposi.test(position(flu), genome.size=1700)

        Monte-Carlo test

        Call: as.randtest(sim = sim, obs = obs, alter = "less")

        Observation: 2

        Based on 999 replicates

        Simulated p-value: 0.497

        Alternative hypothesis: less

          Std.Obs Expectation    Variance

        -0.9813828   2.4774775   0.2367167
```

مقدار بزرگ p-value نشان‌دهنده‌ی عدم انحراف از توزیع یکنواخت می‌باشد.

در اینجا مجدداً تأکید می‌شود همانطور که قبلاً اشاره شد در اشیاء genlight فقط جایگاه‌های SNP دو آللی حفظ می‌شود. بنابراین نتایج آزمون‌هایی مانند فوق، جایگاه‌های چند آللی را شامل نمی‌شوند، هرچند که اطلاعات از دست رفته بدین طریق، محدود است. با استفاده از آرگومان saveNbAlleles می‌توان از تابع fasta2genlight خواست که تعداد آلل‌ها را در هر جایگاه از همردیفی اولیه (و نه صرفاً SNPها)، نگه دارد که در این صورت اطلاعات مربوطه در جزء nb.all.per.loc از بخش other در شیء genlight ذخیره خواهد شد:

```
> flu <- fasta2genlight(myPath, chunk=10,saveNbAlleles=TRUE, quiet=TRUE, parallel=FALSE)

> flu

        /// GENLIGHT OBJECT /////////

        // 80 genotypes,  274 binary SNPs, size: 75.3 Kb

        26 (0.12 %) missing data

        // Basic content

          @gen: list of 80 SNPbin

          @ploidy: ploidy of each individual  (range: 1-1)
```

```
// Optional content
 @ind.names:  80 individual labels
 @loc.all:  274 alleles
 @position: integer storing positions of the SNPs
 @other: a list containing: nb.all.per.loc
```

```
> head(other(flu)$nb.all.per.loc, 20)
 [1] 1 1 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1
```

با استفاده از دستورات زیر می‌توان جایگاه‌های دارای بیش از دو آلل و تعداد آنها را بدست آورد:

```
> which(unlist(other(flu))>2)
```

nb.all.per.loc55	nb.all.per.loc91	nb.all.per.loc93	nb.all.per.loc144
55	91	93	144
nb.all.per.loc156	nb.all.per.loc393	nb.all.per.loc399	nb.all.per.loc410
156	393	399	410
nb.all.per.loc414	nb.all.per.loc432	nb.all.per.loc452	nb.all.per.loc458
414	432	452	458
nb.all.per.loc565	nb.all.per.loc605	nb.all.per.loc704	nb.all.per.loc711
565	605	704	711
nb.all.per.loc724	nb.all.per.loc726	nb.all.per.loc858	nb.all.per.loc958
724	726	858	958
nb.all.per.loc987	nb.all.per.loc1130	nb.all.per.loc1212	nb.all.per.loc1254
987	1130	1212	1254
nb.all.per.loc1314	nb.all.per.loc1392	nb.all.per.loc1560	nb.all.per.loc1578
1314	1392	1560	1578
nb.all.per.loc1633			
1633			

```
> length(which(unlist(other(flu))>2))
 [1] 29
```

نتایج فوق نشان می‌دهد که ۲۹ جایگاه با بیش از دو آلل در قطعه همردیفی مورد بررسی، وجود دارد. با استفاده از اطلاعات مربوطه در جزء nb.all.per.loc از بخش other در شیء genlight، می‌توان میزان پلی‌مورفیسم را در قطعه ژنومیِ مورد بررسی، محاسبه نمود:

```
> 100*mean(unlist(other(flu))>1)
 [1] 17.81305
```

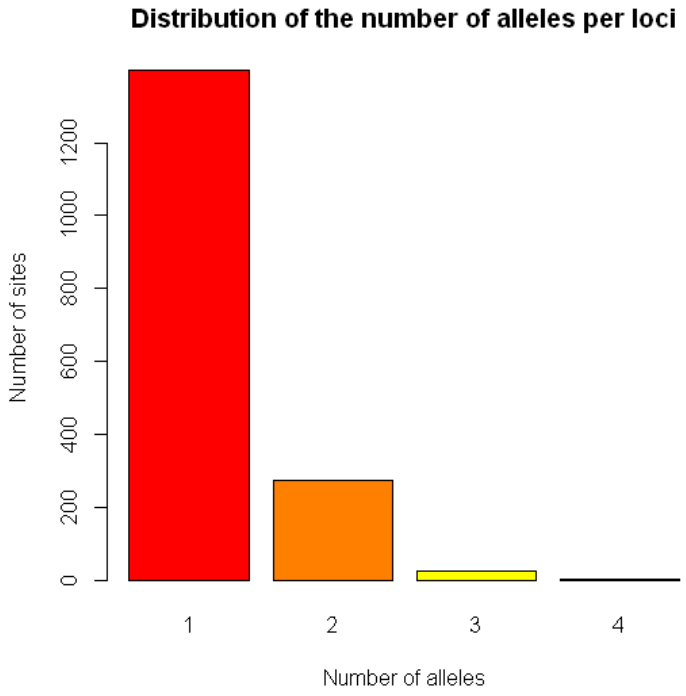
براساس این نتایج، حدود ۱۸ درصد از جایگاه‌ها در همردیفی مورد بررسی، پلی‌مورفیک می‌باشند، که نسبتاً زیاد است. این نتیجه چندان عجیب نیست زیرا که قطعه HA ویروس آنفولانزا به بالا بودن نرخ جهش آن، معروف می‌باشد.

با استفاده از تابع table، می‌توان تعداد جایگاه‌های یک، دو، یا چند آللی را در همردیفی اولیه، به تفکیک مشخص نمود و براساس آن نمودار ستونی رسم نمود (شکل ۱۱-۱۷):

```
> table(unlist(other(flu)))
```

1	2	3	4
1398	274	25	4

```
> temp <- table(unlist(other(flu)))
> barplot(temp, main="Distribution of the number of alleles per loci", xlab="Number of alleles", ylab="Number of sites", col=heat.colors(4))
```



شکل ۱۱-۱۷ – نمودار فراوانی تعداد آلل SNPها در قطعه توالی هماگلوتنین (HA) از ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعهH3N2)

نمودار فوق نشان می‌دهد که اکثر جایگاه‌های پلی‌مورفیک، دو آللی می‌باشند. با استفاده از اطلاعات فوق می‌توان میزان از دست رفتن اطلاعات را به آسانی محاسبه نمود:

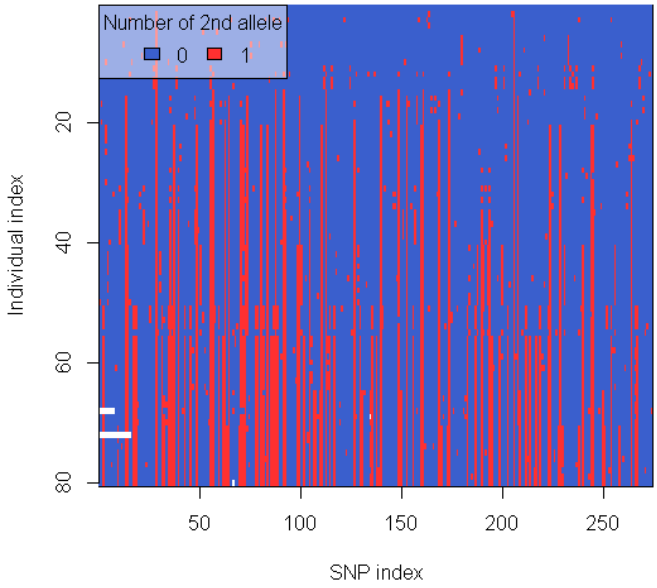
```
> temp <- temp[-1]
> temp <- 100*temp/sum(temp)
> round(temp,1)
   2    3   4
90.4  8.3 1.3
```

نتایج فوق نشان می‌دهد که ۹۰/۴ درصد از جایگاه‌های پلی‌مورفیک، دو آللی بوده‌اند و مابقی آنها عمدتاً سه آللی می‌باشند و در مجموع ۹/۶ درصد از اطلاعات از دست رفته است. باید توجه داشت که این نتایج با توجه به نرخ جهش‌زایی بالای قطعه HA، استثنائی می‌باشد و میزان از دست رفتن اطلاعات، معمولاً کمتر از این مقدار است.

تجزیه‌های پایه روی داده‌های انبوه

در شروع تجزیه داده‌های انبوه بهتر است با ترسیم نمودار، ابتدا تصویر کلی از وضعیت جایگاه‌های SNP در افراد مورد مشاهده قرار گیرد (شکل ۱۱-۱۸). این کار با استفاده از تابع glPlot که قبلاً توضیح داده شد، قابل انجام است:

```
> library(adegenet)
> myPath <- system.file("files/usflu.fasta",package="adegenet")
> flu <- fasta2genlight(myPath, chunk=10, parallel=FALSE)
> glPlot(flu, posi="topleft")
```

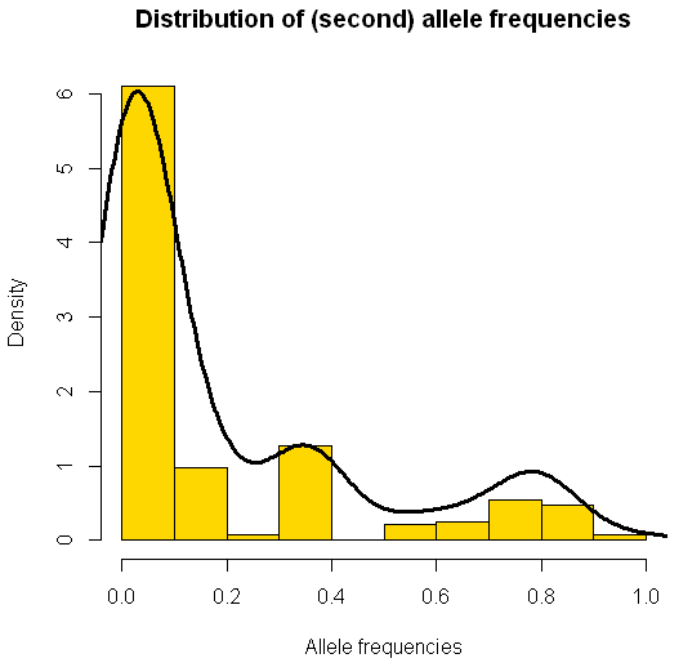


شکل ۱۱-۱۸ – نمودار فراوانی دومین آلل از جایگاه‌های SNP دو آللی در قطعه توالی هماگلوتنین (HA) مربوط به ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2)

نوارهای سفید در نمودار که حدوداً در اطراف فرد شماره ۷۰ مشاهده می‌شود، مربوط به مربوط به داده‌های گمشده سی SNP اولیه است.

با استفاده از تابع glMean که قبلاً شرح داده شد، محاسبه میانگین فراوانی تعداد آلل دوم امکان پذیر است. سپس می‌توان با استفاده از تابع hist، نمودار توزیع فراوانی آلل دوم را رسم نمود (شکل ۱۱-۱۹):

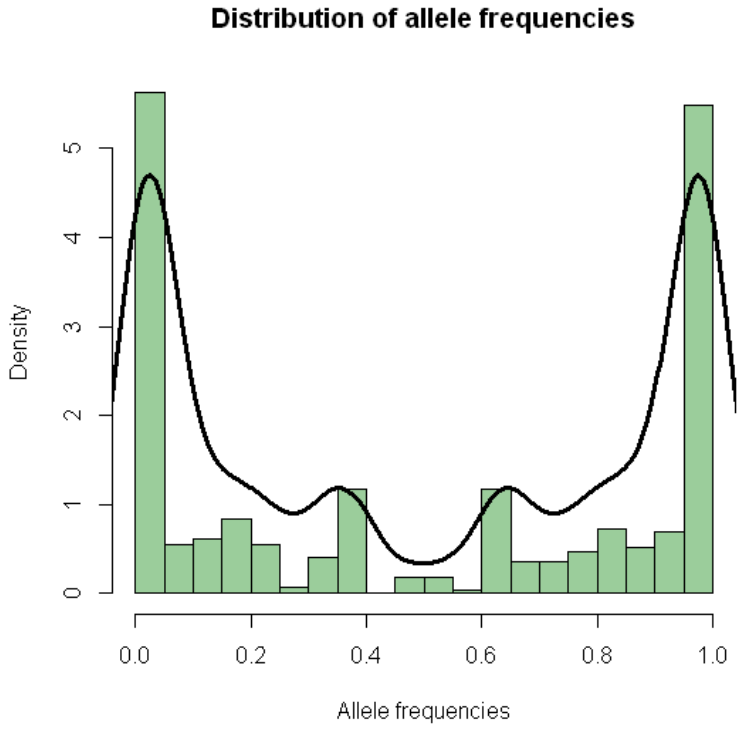
```
> myFreq <- glMean(flu)
> hist(myFreq, proba=TRUE, col="gold", xlab="Allele frequencies", main="Distribution of (second) allele frequencies")
> temp <- density(myFreq)
> lines(temp$x, temp$y*1.8,lwd=3)
```



شکل ۱۱-۱۹ – نمودار فراوانی نسبی دومین آلل از جایگاه‌های SNP دو آللی در قطعه توالی هماگلوتنین (HA) مربوط به ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2)

در لوکوس‌های دو آللی، اطلاعات یک آلل به تنهایی کافی است و اطلاعات آلل دیگر اضافه محسوب می‌شود، مثلاً از روی فراوانی نسبی یک آلل، فراوانی نسبی آلل دیگر قابل محاسبه است. به همین دلیل عموماً تجزیه‌ها برمبنای یکی از آلل‌ها انجام می‌گیرد. اما استثنائاً در مورد نمودار توزیع فراوانی آلل‌ها، چنانچه فراوانی هر دو آلل در جایگاه‌های SNP مدنظر قرار گیرد، نتایج قابل تفسیر و واضح‌تری بدست می‌آید (شکل ۱۱-۲۰):

```
> myFreq <- glMean(flu)
> myFreq <- c(myFreq, 1-myFreq)
> hist(myFreq, proba=TRUE, col="darkseagreen3", xlab="Allele frequencies", main="Distribution of allele frequencies", nclass=20)
> temp <- density(myFreq, bw=.05)
> lines(temp$x, temp$y*2,lwd=3)
```



شکل ۱۱-۲۰ – نمودار فراوانی نسبی آلل‌های جایگاه‌های SNP دو آللی در قطعه توالی هماگلوتنین (HA) مربوط به ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2)

نمودار فوق نشان می‌دهد که تعداد زیادی از آلل‌ها در هر دو جهت در حال تثبیت شدن هستند (فراوانی‌های نسبی نزدیک به صفر یا یک)، اما در عین حال بخش قابل توجهی نیز دارای فراوانی نسبی بینابینی هستند که می‌توانند حاوی اطلاعات زیستی مهمی باشند.

محاسبه و ترسیم فواصل ژنتیکی در داده‌های انبوه

با استفاده از تابع bitwise.dist در پکیج poppr می‌توان تفاوت‌های آللی بین افراد را در اشیاء genlight بدست آورد. به عنوان مثال ابتدا با استفاده از تابع glSim، تعداد پانصد SNP غیرساختاری و پانصد SNP ساختاری را در ۱۰ فرد دیپلوئید، شبیه‌سازی کرده و فواصل ژنتیکی بین آنها را توسط تابع bitwise.dist بدست می‌آوریم:

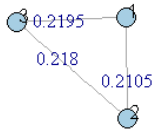
```
> library(poppr)
```

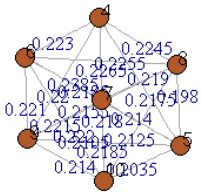
```
> library(adegenet)
> set.seed(999)
> x <- glSim(n.ind = 10, n.snp.nonstruc = 5e2, n.snp.struc = 5e2, ploidy = 2)
> xd <- bitwise.dist(x)
> xd
```

	1	2	3	4	5	6	7	8	9
2	0.2105								
3	0.2195	0.218							
4	0.395	0.3825	0.3855						
5	0.4075	0.391	0.396	0.2175					
6	0.392	0.3855	0.3985	0.223	0.2185				
7	0.3955	0.392	0.391	0.2265	0.214	0.2385			
8	0.3955	0.383	0.397	0.2245	0.198	0.2255	0.219		
9	0.377	0.3805	0.3945	0.22	0.2105	0.221	0.2215	0.2155	
10	0.398	0.3925	0.4005	0.218	0.2035	0.222	0.2185	0.2125	0.214

به همان ترتیب با استفاده از تابع gengraph در پکیج adegenet می‌توان فواصل ژنتیکی بین افراد را ترسیم نمود (شکل ۱۱-۲۱):

```
> xg<-gengraph(xd, cutoff=0.25)
> plot(xg$graph)
```





شکل ۱۱-۲۱ – نمایش فواصل ژنتیکی در ده فرد دیپلوئید شبیه‌سازی شده برای پانصد SNP غیرساختاری و پانصد SNP ساختاری

تجزیه به مؤلفه‌های اصلی (glpca)

با استفاده از تابع glPca می‌توان تجزیه به مؤلفه‌های اصلی را برای اشیاء genlight انجام داد:

glPca(x, center = TRUE, scale = FALSE, nf = NULL)

x: یک شیء genlight است.

center: گزاره‌ای منطقی است که مشخص می‌کند آیا اعداد مربوط به شمارش آلل‌ها، کانونی (centered) شود یا خیر، این گزاره بطور پیش‌فرض، TRUE قرار داده شده است.

scale: گزاره‌ای منطقی است که مشخص می‌کند آیا اعداد مربوط به شمارش آلل‌ها، مقیاس (scaled) شود یا خیر، این گزاره بطور پیش‌فرض، FALSE قرار داده شده است.

nf: عددی است که نشان‌دهندهِ تعداد مؤلفه‌های اصلی است که بایستی نگه داشته شود. چنانچه مقدار آن NULL باشد، یک نمودار screeplot از مقادیر ویژه (eigenvalues) ظاهر می‌شود و از کاربر می‌خواهد که تعداد محورهایی که لازم است نگه داشته شود را مشخص نماید.

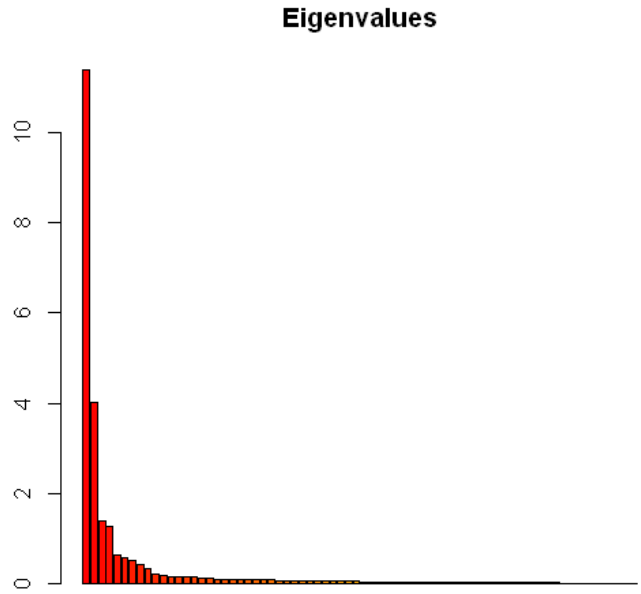
خروجی تابع glPca، "لیستی" است که متشکل از وکتور عددی از مقادیر ویژه (eig)، ماتریس مؤلفه‌های اصلی (scores)، و ماتریس ضرایب (loadings) می‌باشد. ماتریس مؤلفه‌های اصلی (scores)، شامل مختصات هر فرد (ردیف) و هر مؤلفه اصلی (ستون) می‌باشد و ماتریس ضرایب (loadings)، ضرایب هر SNP (در ردیف) و هر مؤلفه اصلی (در ستون) را در بردارد.

به عنوان مثال، برای مجموعه داده‌های آنفولانزا تجزیه به مؤلفه‌های اصلی بصورت زیر خواهد بود:

توجه: آرگومان parallel=FALSE سبب "عدم" پردازش موازی چندهسته‌ای، می‌شود. توجه داشته باشید که پردازش موازی بر روی سیستم Windows پشتیبانی نمی‌شود. چنانچه بخواهید پردازش داده‌ها در دستگاه رایانه شما بصورت چندهسته‌ای انجام شود، می‌توانید آرگومان parallel=TRUE را انتخاب کنید که در اینصورت باید از قبل پکیج parallel، مربوط به پردازش موازی چند هسته‌ای را نصب نموده باشید. همچنین توجه داشته باشید که اگر تعداد مؤلفه‌های اصلی که باید نگه داشته شوند توسط آرگومان nf تعیین نشده باشد، نمودار مقادیر ویژه ظاهر شده (شکل ۱۱-۲۲) و از کاربر خواسته می‌شود که تعداد مذکور را تعیین نماید در غیر اینصورت، تابع glPca با تعداد مؤلفه اصلی مشخص شده در آرگومان nf عمل خواهد نمود. تفاوت نتایج حاصل از اجرای دو تابع glPca را در دستورات زیر مقایسه نمایید:

```
> library(adegenet)
> myPath <- system.file("files/usflu.fasta",package="adegenet")
```

```
> flu <- fasta2genlight(myPath, chunk=10, parallel=FALSE)
> pca1 <- glPca(flu, nf=2, parallel=FALSE)
> pca1 <- glPca(flu, parallel=FALSE)
     Select the number of axes: 2
```



شکل ۱۱-۲۲ – نمودار ستونی مقادیر ویژه در تجزیه به مؤلفه‌های اصلی جایگاه‌های SNP در قطعه توالی هماگلوتنین (HA) مربوط به ۸۰ ایزوله ویروس آنفولانزای فصلی (مجموعه H3N2)

شیء ایجاد شده (pca1)، لیستی متشکل از اجزای مختلف، مربوط به نتایج تجزیه به مؤلفه‌های اصلی می‌باشد:

```
> pca1
     === PCA of genlight object ===
     Class: list of type glPca
     Call ($call):glPca(x = flu, nf = 2, parallel = FALSE)
     Eigenvalues ($eig):
      11.385 4.019 1.391 1.275 0.636 0.569 ...
```

Principal components ($scores):

matrix with 80 rows (individuals) and 2 columns (axes)

Principal axes ($loadings):

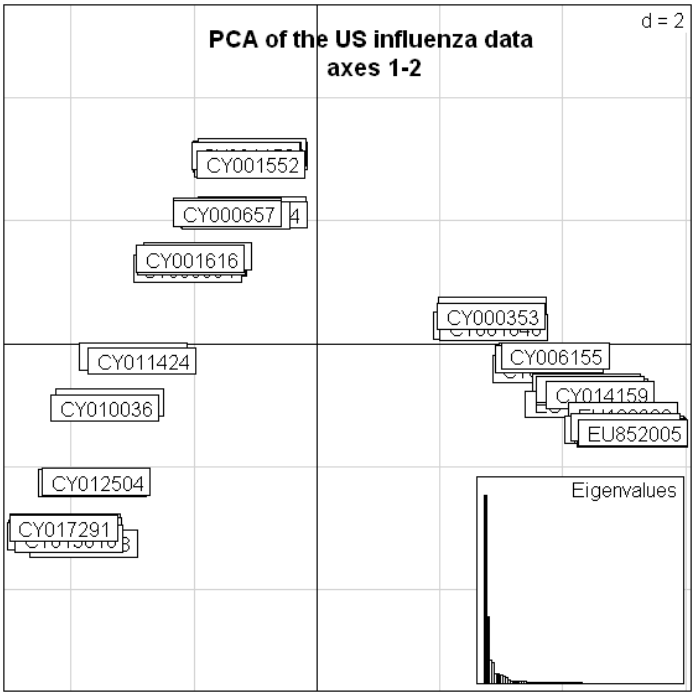
matrix with 274 rows (SNPs) and 2 columns (axes)

```
> names(pca1)
```

[1] "eig" "scores" "loadings" "call"

با استفاده از تابع scatterplot بر روی شیء glPca، می‌توان نمودار پراکنش افراد مبتنی بر مؤلفه‌های اصلی را ترسیم نمود (شکل ۱۱-۲۳):

```
> scatter(pca1, posi="bottomright")
> title("PCA of the US influenza data\n axes 1-2")
```



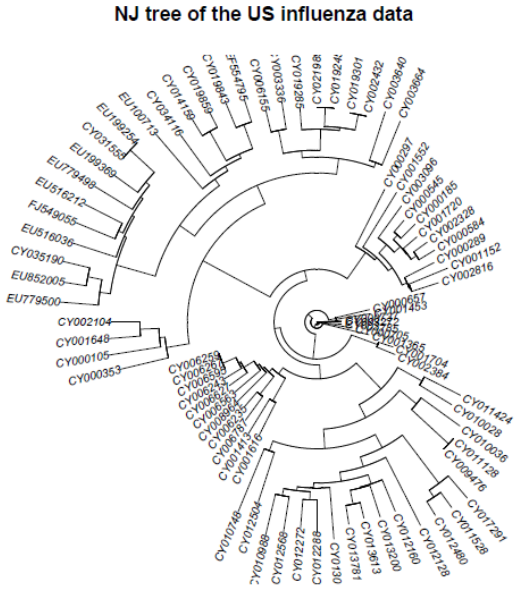
شکل ۱۱-۲۳ – تکفکیک ایزوله‌های ویروس آنفولانزای فصلی براساس مؤلفه‌های اصلی مربوط به جایگاه‌های SNP در قطعه توالی هماگلوتنین (HA)

در این نمودار، ایزوله‌ها براساس مؤلفه اصلی اول به دو گره تقسیم شده‌اند، ولی مؤلفه اصلی دوم، صف آرایی ایزوله‌ها را در امتداد شیبی از افتراق ژنتیکی نشان می‌دهد.

ترسیم درخت فیلوژنی

با ترسیم درخت مبتنی بر روش اتصال همسایه‌ها (شکل ۱۱-۲۴) می‌توان ساختار بدست آمده از تجزیه به مؤلفه‌های اصلی را مورد تأیید قرار داد (ادامه دستوارت از بخش قبل):

```
> library(ape)
> tre <- nj(dist(as.matrix(flu)))
> tre

    Phylogenetic tree with 80 tips and 78 internal nodes.

    Tip labels:
        CY013200, CY013781, CY012128, CY013613, CY012160, CY012272, ...
    Unrooted; includes branch lengths.

> plot(tre, typ="fan", cex=0.7)
> title("NJ tree of the US influenza data")
```

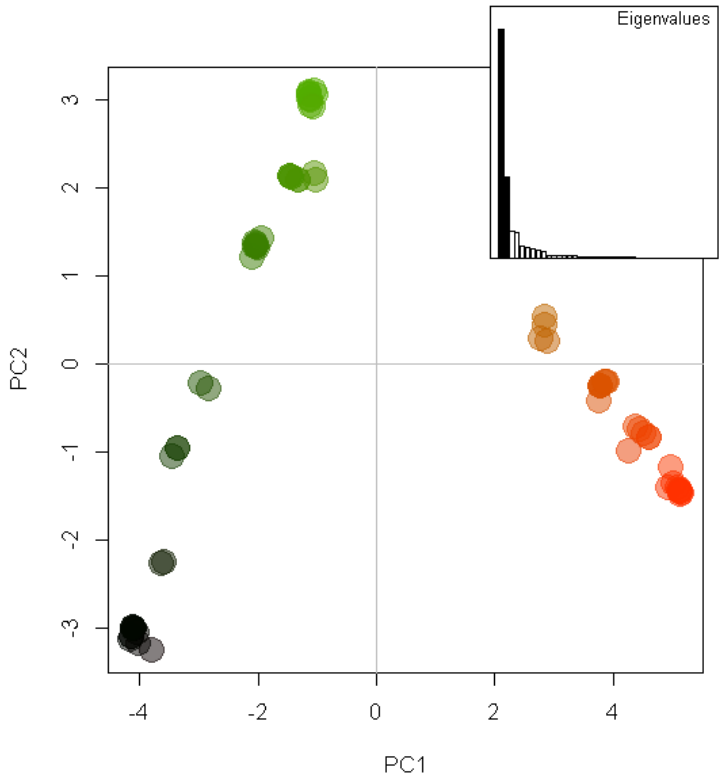


شکل ۱۱-۲۴ – درخت فیلوژنی ایزوله‌های ویروس آنفولانزای فصلی براساس جایگاه‌های SNP در قطعه توالی هماگلوتنین (HA)

تطابق نمودار مؤلفه‌های اصلی و درخت فیلوژنی

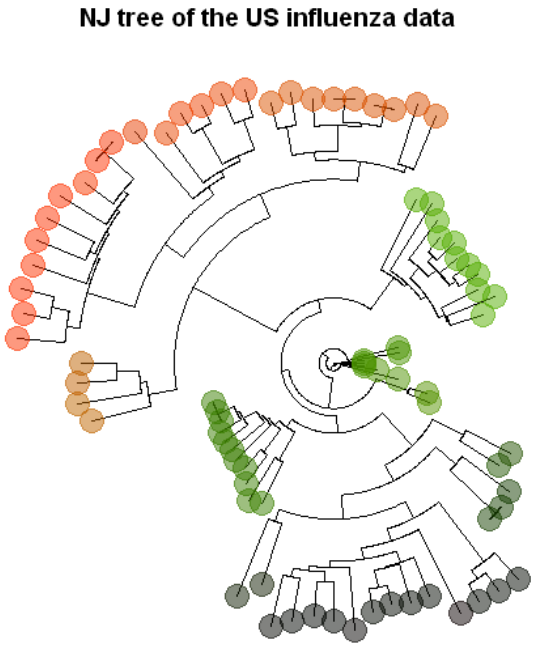
نتایج تجزیه مؤلفه‌های اصلی و درخت فیلوژنی با اختصاص رنگ بجای حروف، بیشتر قابل تطابق خواهند شد (شکل‌های ۱۱-۲۵ و ۱۱-۲۶). بدین منظور می‌توان از تابع colorplot استفاده نمود (ادامه از بخش قبل):

```
> myCol <- colorplot(pca1$scores,pca1$scores, transp=TRUE, cex=4)
> abline(h=0,v=0, col="grey")
> add.scatter.eig(pca1$eig[1:40],2,1,2, posi="topright", inset=.05, ratio=.3)
```



شکل ۱۱-۲۵ – نمودار پراکنش ایزوله‌های ویروس آنفولانزای فصلی براساس مؤلفه‌های اصلی مربوط به جایگاه‌های SNP در قطعه توالی هماگلوتنین (HA) با استفاده از رنگ بجای حروف

```
> plot(tre, typ="fan", show.tip=FALSE)
> tiplabels(pch=20, col=myCol, cex=4)
> title("NJ tree of the US influenza data")
```



شکل ۱۱-۲۶ – درخت فیلوژنی ایزوله‌های ویروس آنفولانزای فصلی براساس جایگاه‌های SNP در قطعه توالی هماگلوتنین (HA) با استفاده از رنگ بجای حروف

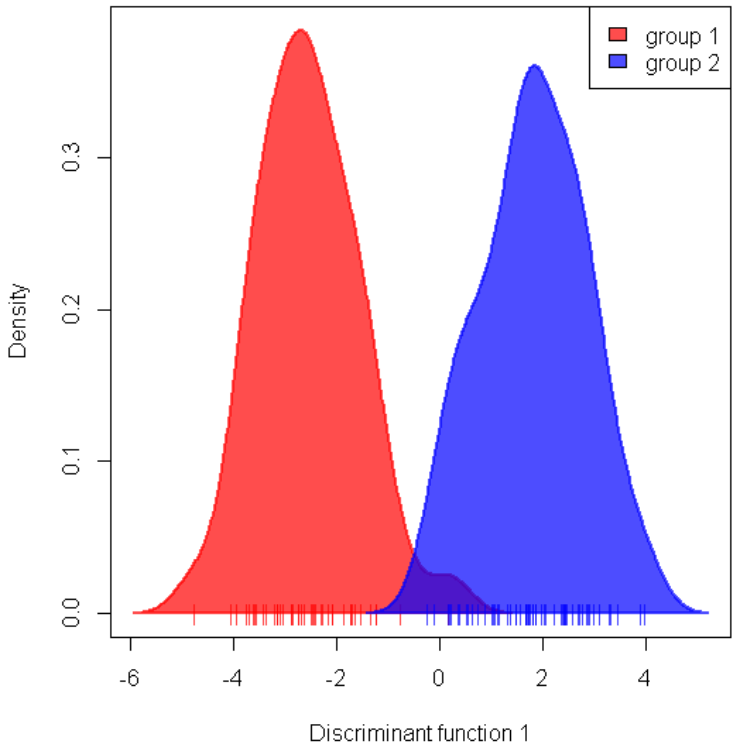
براساس نمودارهای فوق می‌توان گفت نتایج تجزیه به مؤلفه‌های اصلی و درخت فیلوژنی با یکدیگر در تطابق بوده و تکمیل‌کننده یکدیگر می‌باشند. درخت فیلوژنی گروه‌های ایزوله‌ها را بهتر متمایز ساخته است، ولی شیب افتراق ژنتیکی، توسط تجزیه به مؤلفه‌های اصلی به نحو واضح‌تری نشان داده شده است.

تجزیه تشخیصی مبتنی بر مؤلفه‌های اصلی (DAPC)

تجزیه DAPC همانطور که در بخش اشیاء genind توضیح داده شد، با استفاده از تابع dapc بر روی اشیاء genlight نیز قابل اجرا می‌باشد. به منظور تشریح کارایی این تجزیه بر روی اشیاء genlight به عنوان مثال، ابتدا با استفاده از تابع glSim، تعداد ده‌هزار SNP غیرساختاری و پنجاه SNP ساختاری را در ۱۰۰ فرد، شبیه‌سازی کرده و سپس تجزیه DAPC را انجام می‌دهیم (شکل ۱۱-۲۷):

```
> library(adegenet)
```

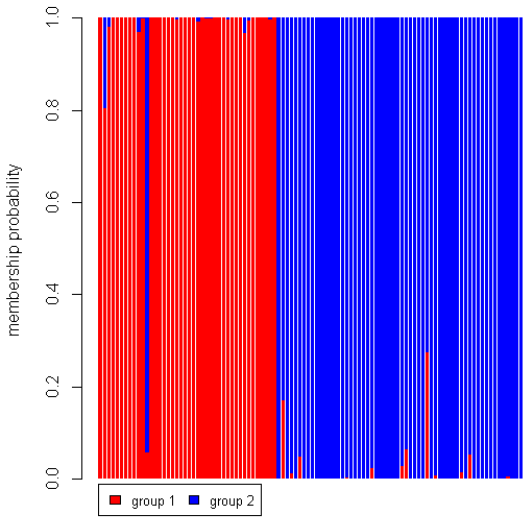
```
> x <- glSim(100, 1e4, 50)

> dapc1 <- dapc(x, n.pca=10, n.da=1, parallel=FALSE)

> scatter(dapc1,scree.da=FALSE, bg="white", posi.pca="topright", legend=TRUE, txt.leg=paste("group", 1:2), col=c("red","blue"))
```



شکل ۱۱-۲۷ – تمایز دو گروه توسط نمودار توزیع چگالی مبتنی بر تابع اول تجزیه DAPC در ۱۰۰ فرد شبیه‌سازی شده برای ده‌هزار SNP غیرساختاری و پنجاه SNP ساختاری

همانطور که در نمودار مشخص است با استفاده از تجزیه DAPC دو گروه بخوبی (به استثناء مقداری جزئی از همپوشانی) از یکدیگر تفکیک شده‌اند. تفکیک دو گروه و افراد اختلاط یافته را می‌توان با تابع compoplot بهتر نشان داد (شکل ۱۱-۲۸):

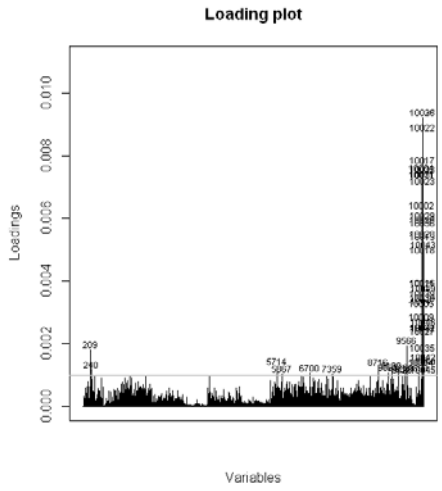
```
> compoplot(dapc1, col=c("red","blue"),lab="", txt.leg=paste("group", 1:2), ncol=2)
```



شکل ۱۱-۲۸ – تفکیک دو گروه و شناسایی افراد اختلاط یافته براساس احتمال عضویت در گروه‌ها پس از انجام تجزیه DAPC روی ۱۰۰ فرد شبیه‌سازی شده برای ده‌هزار SNP غیرساختاری و پنجاه SNP ساختاری

همچنین آلل‌هایی که بیشترین نقش در تمایز دارند را می‌توان با استفاده از تابع loadingplot شناسایی نمود (شکل ۱۱-۲۹):

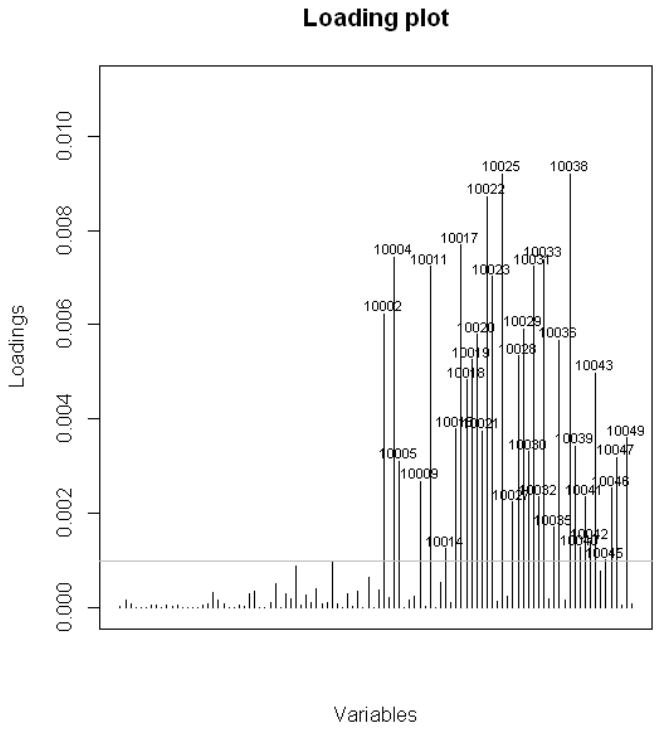
```
> loadingplot(dapc1$var.contr, thres=1e-3)
```



شکل ۱۱-۲۹ – شناسایی آلل‌های دارای بیشترین نقش در تمایز بین گروه‌ها مبتنی بر تجزیه DAPC روی ۱۰۰ فرد شبیه‌سازی شده برای ده‌هزار SNP غیرساختاری و پنجاه SNP ساختاری

می‌توان نمودار ضرایب را محدود به صد SNP آخری نمود تا نقش پنجاه SNP ساختاری که در فرایند شبیه‌سازی تعریف شد، بهتر مشخص شود (شکل ۱۱-۳۰):

```
> loadingplot(tail(dapc1$var.contr[,1],100), thres=1e-3)
```



شکل ۱۱-۳۰ – شناسایی آلل‌های دارای بیشترین نقش در تمایز بین گروه‌ها مبتنی بر تجزیه DAPC روی ۱۰۰ فرد شبیه‌سازی شده برای ده‌هزار SNP غیرساختاری و پنجاه SNP ساختاری با اعمال محدودیت جهت شمول صد SNP ساختاری انتهایی

برای شناسایی آلل‌های SNP که بیشترین مشارکت را در ساختار فنوتیپی دارند، همچنین می‌توان از تابع snpzip استفاده نمود. این رویه از تجزیه تشخیصی مؤلفه‌های اصلی (DAPC) برای تعیین میزان سهم آلل‌ها در تفکیک جمعیت‌ها از یکدیگر استفاده می‌نماید. سپس میزان مشارکت در DAPC به عنوان معیاری از فواصل بین آلل‌ها در نظر گرفته می‌شود و از خوشه‌بندی سلسله مراتبی، برای شناسایی دو گروه آلل‌ها شامل SNPهای ساختاری و SNPهای غیرساختاری استفاده می‌شود:

snpzip(snps, y, plot = TRUE, xval.plot = FALSE, loading.plot = FALSE, method = c("complete", "single", "average", "centroid", "mcquitty", "median", "ward"))

snps: ماتریس SNPها که به عنوان ورودی استفاده می‌شود.

y: فاکتوری که تعیین کننده عضویت گروهی افراد است و یا یک شیء dapc.

plot: یک گزاره منطقی که مشخص می‌سازد آیا نمایش گرافیکی نتایج DAPC نشان داده شود یا خیر.

xval.plot: یک گزاره منطقی که مشخص می‌سازد آیا مرحله cross-validation نمایش داده شود یا خیر.

loading.plot: یک گزاره منطقی که مشخص می‌سازد آیا نمودار ضرایب (loading plot) مشتمل بر آستانه گزینش SNPها از لحاظ میزان مشارکت در توابع تشخیصی، نمایش داده شود یا خیر.

method: روش خوشه‌بندی

مقادیر خروجی تابع snpzip "لیستی" متشکل از چهار جزء (اگر y، یک فاکتور باشد) یا دو جزء (اگر y ، یک شیء dapc باشد) به شرح زیر می‌باشد:

- اولین جزء، تعداد مؤلفه‌های اصلی (PCs) نگه داشته شده در DAPC را نشان می‌دهد.

- دومین جزء، لیستی است که اولاً تعداد SNPهای ساختاری و SNPهای غیرساختاری شناسایی شده توسط تابع snpzip را نشان می‌دهد، ثانیاً لیستی از آلل‌های ساختاری ارائه می‌دهد، ثالثاً اسامی آلل‌های انتخاب شده را مشخص می‌سازد و رابعاً جزئیات مربوط به میزان مشارکت این آلل‌های ساختاری را در DAPC ارائه می‌دهد.

- سومین جزء، معیاری از موفقیت تجزیه تشخیصی بطور کلی و هم برحسب گروه را نشان می‌دهد.

- چهارمین جزء، یک شیء از نوع DAPC است، چنانچه y، یک فاکتور، بوده باشد.

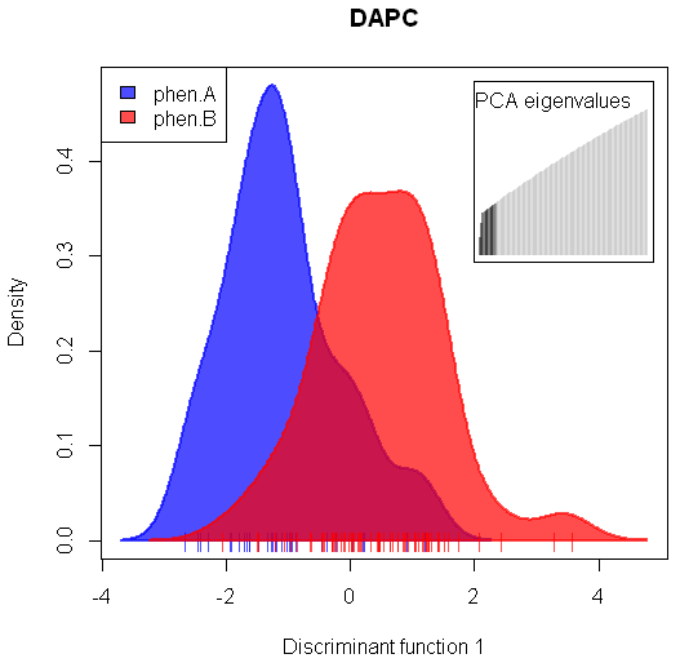
به عنوان مثال، ابتدا با استفاده از تابع glSim تعداد ده هزار SNP غیرساختاری و ده SNP ساختاری مستقل (LD = FALSE) را در ۱۰۰ فرد، در قالب چهار جمعیت اجدادی (k = 4) به نحوی شبیه‌سازی می‌کنیم که ۱۰۰ فرد مذکور به نسبت ۷ به ۳ در دو گروه، با تفاوت بین گروهی نسبتاً قوی (alpha = 0.4) واقع شوند:

```
> simpop <- glSim(100, 10000, n.snp.struc = 10, grp.size = c(0.3,0.7),  LD = FALSE, alpha = 0.4, k = 4)
> class(simpop)
        [1] "genlight"
        attr(,"package")
```

[1] "adegenet"

خروجی تابع glSim، شیئی از نوع genlight است. برای اجرای تابع snpzip نیاز به شیء ماتریس داریم:

```
> snps <- as.matrix(simpop)
> phen <- simpop@pop
> outcome <- snpzip(snps, phen, method = "centroid")
```



شکل ۱۱-۳۱ – تمایز دو گروه توسط نمودار توزیع چگالی مبتنی بر تابع اول تجزیه DAPC در ۱۰۰ فرد شبیه‌سازی شده برای ده‌هزار SNP غیرساختاری و ده SNP ساختاری مستقل و در نظر گرفتن چهار جمعیت اجدادی

```
> outcome[[1]]
[1] "20"
> outcome[[2]]
$`Number of selected vs. unselected alleles`
[1]    10 10000

$`List of selected alleles`
 [1] 10001 10002 10003 10004 10005 10006 10007 10008 10009 10010

$`Names of selected alleles`
NULL

$`Contributions of selected alleles to discriminant axis`
    V10001          V10002          V10003          V10004          V10005          V10006
0.006306032     0.006864486     0.005203368     0.007364668     0.006468866     0.007102299
    V10007          V10008          V10009          V10010
```

```
0.005279288         0.005035686    0.006060378    0.004656963

> outcome[[3]]
        [1] 0.9300000 0.9246947
> outcome[[4]]
              ################################################
              # Discriminant Analysis of Principal Components #
              ################################################
        class: dapc
        $call: dapc.data.frame(x = as.data.frame(x), grp = ..1, n.pca = ..2,
          n.da = ..3)

        $n.pca: 20 first PCs of PCA used
        $n.da: 1 discriminant functions saved
        $var (proportion of conserved variance): 0.436

        $eig (eigenvalues): 198.6  vector    length content
        1 $eig      1      eigenvalues
        2 $grp      100    prior group assignment
        3 $prior    2      prior group probabilities
        4 $assign   100    posterior group assignment
        5 $pca.cent 10010  centring vector of PCA
        6 $pca.norm 10010  scaling vector of PCA
        7 $pca.eig  99     eigenvalues of PCA

          data.frame    nrow  ncol content
        1 $tab          100   20   retained PCs of PCA
        2 $means        2     20   group means
        3 $loadings     20    1    loadings of variables
        4 $ind.coord    100   1    coordinates of individuals (principal components)
        5 $grp.coord    2     1    coordinates of groups
        6 $posterior    100   2    posterior membership probabilities
        7 $pca.loadings 10010 20   PCA loadings of original variables
        8 $var.contr    10010 1    contribution of original variables
```

براساس نمودار (شکل ۱۱-۳۱) مشاهده می‌شود که تابع تشخیصی، دو گروه افراد را از یکدیگر متمایز نموده است و براساس مقادیر حاصل شده در نتایج تجزیه فوق، تعداد ده آلل SNP با بیشترین مشارکت در محور تجزیه تشخیصی، شناسایی شده است.

فهرست منابع

Agapow, P.M. and Burt, A., 2001.Indices of multilocus linkage disequilibrium. *Molecular Ecology Notes*, *1*(1-2), pp.101-102.

Akey, J.M., Zhang, G., Zhang, K., Jin, L. and Shriver, M.D., 2002.Interrogating a high-density SNP map for signatures of natural selection. *Genome research*, *12*(12), pp.1805-1814.

Akter, S., Huq, M.A., Jung, Y.J., Cho, Y.G. and Kang, K.K., 2016.Application of Single Nucleotide Polymorphism Markers for Selection of Male Sterility in Crop Plants. *Plant Breeding and Biotechnology*, *4*(4), pp.379-386.

Alkan, C., Coe, B.P. and Eichler, E.E., 2011. Genome structural variation discovery and genotyping. *Nature Reviews Genetics*, *12*(5), pp.363-376.

Allendorf, F.W. and Luikart, G., 2007. Conservation and the genetics of populations. *Mammalia*, *2007*, pp.189-197.

Allendorf, F.W. and Luikart, G., 2009. *Conservation and the genetics of populations*.John Wiley & Sons.

Amato, G., DeSalle, R., Ryder, O.A. and Rosenbaum, H.C. eds., 2009.*Conservation genetics in the age of genomics*.Columbia University Press.

Anderson, A.D. and Weir, B.S., 2007. A maximum-likelihood method for the estimation of pairwise relatedness in structured populations. *Genetics*, *176*(1), pp.421-440.

Avise, J.C., 1994. Molecular markers, natural history and evolution.Chapman & Hall, London.

Avise, J.C., 2012. *Molecular markers, natural history and evolution*.Springer Science & Business Media.

Ayala, F.J., 1982. *Population and evolutionary genetics: a primer*. The Benjamin/Cummings Publishing Company, Inc. USA.

Belkhir, K., Borsa, P., Chikhi, L., Raufaste, N. and Bonhomme, F., 1996. GENETIX 4.05, logiciel sous Windows TM pour la génétique des populations. *Laboratoiregénome, populations, interactions, CNRS UMR*, *5000*, pp.1996-2004.

Benzécri, J.P., 1992. *Correspondence analysis handbook*. Marcel Dekker.

Botstein, D., White, R.L., Skolnick, M. and Davis, R.W., 1980. Construction of a genetic linkage map in man using restriction fragment length polymorphisms. *American journal of human genetics*, *32*(3), p.314.

Brown, A.H.D., Feldman, M.W. and Nevo, E., 1980. Multilocus structure of natural populations of *Hordeumspontaneum*. *Genetics*, *96*(2), pp.523-536.

Buttigieg, P.L. and Ramette, A., 2014. A guide to statistical analysis in microbial ecology: a community-focused, living review of multivariate data analyses. *FEMS microbiology ecology*, *90*(3), pp.543-550.

Carothers, A.D., Rudan, I., Kolcic, I., Polasek, O., Hayward, C., Wright, A.F., Campbell, H., Teague, P., Hastie, N.D. and Weber, J.L., 2006. Estimating human inbreeding coefficients: comparison of genealogical and marker heterozygosity approaches. *Annals of human genetics*, *70*(5), pp.666-676.

Cavalli-Sforza, L.L. and Edwards, A.W., 1967.Phylogenetic analysis.Models and estimation procedures. *American journal of human genetics*, *19*(3 Pt 1), p.233.

Chapuis, M.P. and Estoup, A., 2007. Microsatellite null alleles and estimation of population differentiation. *Molecular biology and evolution*, *24*(3), pp.621-631.

Charlesworth, B. and Charlesworth, D., 2010. *Elements of evolutionary genetics*. Greenwood Village: Roberts and Company Publishers.

Christiansen, F.B., 1999. *Population genetics of multiple loci*.John Wiley & Sons Ltd.

Chybicki, I.J. and Burczyk, J., 2009. Simultaneous estimation of null alleles and inbreeding coefficients. *Journal of Heredity*, *100*(1), pp.106-113.

Conner, J.K. and Hartl, D.L., 2004. *A primer of ecological genetics*.Sinauer Associates Incorporated.

Corander, J., Waldmann, P. and Sillanpää, M.J., 2003. Bayesian analysis of genetic differentiation between populations. *Genetics*, *163*(1), pp.367-374.

Crow, J.F. and Kimura, M., 1970.An introduction to population genetics theory. *An introduction to population genetics theory.*

Culley, T.M., Wallace, L.E., Gengler-Nowak, K.M. and Crawford, D.J., 2002.A comparison of two methods of calculating G_{ST}, a genetic measure of population differentiation. *American Journal of Botany*, *89*(3), pp.460-465.

Dewar, R.C., Sherwin, W.B., Thomas, E., Holleley, C.E. and Nichols, R.A., 2011. Predictions of single-nucleotide polymorphism differentiation between two populations in terms of mutual information. *Molecular ecology*, *20*(15), pp.3156-3166.

Doolittle, D.P., 2012. *Population Genetics:: Basic Principles* (Vol. 16). Springer Science & Business Media.

Edwards, A.W.F., 1971. Distances between populations on the basis of gene frequencies. *Biometrics*, pp.873-881.Cavalli-Sforza L.L. and Edwards A.W.F. (1967) Phylogenetic analysis: models and estimation procedures.

Everitt, B.S. and Dunn, G., 2001. *Applied multivariate data analysis*.Oxford University Press.

Ewens, W.J., 2004. Mathematical population genetics. I. Theoretical introduction. Interdisciplinary Applied Mathematics, 27.

Ewens, W.J., 2012. *Mathematical population genetics 1: theoretical introduction* (Vol. 27). Springer Science & Business Media.

Excoffier, L. and Heckel, G., 2006. Computer programs for population genetics data analysis: a survival guide. *Nature Reviews Genetics*, *7*(10), pp.745-758.

Excoffier, L., Smouse, P.E. and Quattro, J.M., 1992. Analysis of molecular variance inferred from metric distances among DNA haplotypes: application to human mitochondrial DNA restriction data. *Genetics*, *131*(2), pp.479-491.

Falk, D.A. and Holsinger, K.E., 1991. *Genetics and conservation of rare plants*. Oxford University Press on Demand.

Felsenstein, J. and Churchill, G.A., 1996. A Hidden Markov Model approach to variation among sites in rate of evolution. *Molecular Biology and Evolution*, *13*(1), pp.93-104.

Felsenstein, J., 1981. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of molecular evolution*, *17*(6), pp.368-376.

Foll, M. and Gaggiotti, O., 2008. A genome-scan method to identify selected loci appropriate for both dominant and codominant markers: a Bayesian perspective. *Genetics*, *180*(2), pp.977-993.

Fox, J. and Andersen, R., 2005. Using the R statistical computing environment to teach social statistics courses. *Department of Sociology, McMaster University*, pp.2-4.

Frankham, R., Briscoe, D.A. and Ballou, J.D., 2002. *Introduction to conservation genetics*.Cambridge university press.

Galtier, N. and Gouy, M., 1995.Inferring phylogenies from DNA sequences of unequal base compositions. *Proceedings of the National Academy of Sciences*, *92*(24), pp.11317-11321.

Gerlach, G., Jueterbock, A., Kraemer, P., Deppermann, J. and Harmand, P., 2010. Calculations of population differentiation based on G_{ST} and D: forget G_{ST} but not all of statistics!. *Molecular Ecology*, *19*(18), pp.3845-3852.

Gillespie, J.H., 1994. *The causes of molecular evolution*.Oxford University Press.

Gillespie, J.H., 2010. *Population genetics: a concise guide*. JHU Press.

Good, I.J., 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, pp.237-264.

Goudet, J., 2002. Fstat 2.9.3.2. URL: http://www2.unil.ch/popgen/softwares/fstat.htm.

Goudet, J., Raymond, M., de Meeüs, T. and Rousset, F., 1996.Testing differentiation in diploid populations. *Genetics*, *144*(4), pp.1933-1940.

Gower, J.C. and Legendre, P., 1986. Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification*, *3*(1), pp.5-48.

Gower, J.C., 1966. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, pp.325-338.

Greenacre, M., 1992.Correspondence analysis in medical research.*Statistical methods in medical research*, *1*(1), pp.97-117.

Greenacre, M., 2007. *Correspondence analysis in practice*.CRC press.

Grünwald, N.J., Goodwin, S.B., Milgroom, M.G. and Fry, W.E., 2003.Analysis of genotypic diversity data for populations of microorganisms. *Phytopathology*, *93*(6), pp.738-746.

Gu, X. and Li, W.H., 1996. Bias-corrected paralinear and LogDet distances and tests of molecular clocks and phylogenies under nonstationary nucleotide frequencies. *Molecular biology and evolution*, *13*(10), pp.1375-1383.

Guo, F., Dey, D.K. and Holsinger, K.E., 2009.A Bayesian hierarchical model for analysis of single-nucleotide polymorphisms diversity in multilocus, multipopulation samples. *Journal of the American Statistical Association*, *104*(485), pp.142-154.

Hall, N., Mercer, L., Phillips, D., Shaw, J. and Anderson, A.D., 2012. Maximum likelihood estimation of individual inbreeding coefficients and null allele frequencies. *Genetics research*, *94*(03), pp.151-161.

Hamilton, M., 2011. *Population genetics*.John Wiley & Sons.

Hartl, D.L., 1988. *A primer of population genetics*.Sinauer Associates, Inc..

Hartl, D.L., Clark, A.G. and Clark, A.G., 1997. *Principles of population genetics* (Vol. 116). Sunderland: Sinauer associates.

Haubold, B. and Hudson, R.R., 2000. LIAN 3.0: detecting linkage disequilibrium in multilocus data. *Bioinformatics*, *16*(9), pp.847-849.

Heck, K.L., van Belle, G. and Simberloff, D., 1975. Explicit calculation of the rarefaction diversity measurement and the determination of sufficient sample size. *Ecology*, *56*(6), pp.1459-1461.

Hedrick, P.W., 1999. Perspective: highly variable loci and their interpretation in evolution and conservation. *Evolution*, pp.313-318.

Hedrick, P.W., 2005. A standardized genetic differentiation measure. *Evolution*, *59*(8), pp.1633-1638.

Hedrick, P.W., 2011. *Genetics of populations*.Jones & Bartlett Learning.

Heller, R. and Siegismund, H.R., 2009. Relationship between three measures of genetic differentiation GST, DEST and G'ST: how wrong have we been?. *Molecular Ecology*, *18*(10), pp.2080-2083.

Henry, R.J., 2013. Molecular Markers in Plants.A John Wiley & Sons, Inc., Publication

Hill, M.O. and Gauch, H.G., 1980. Detrended correspondence analysis: an improved ordination technique. *Vegetatio*, *42*(1-3), pp.47-58.

Hill, M.O., 1974. Correspondence analysis: a neglected multivariate method.*Applied statistics*, pp.340-354.

Höglund, J., 2009. *Evolutionary conservation genetics*.Oxford University Press.

Holsinger, K.E. and Weir, B.S., 2009. Genetics in geographically structured populations: defining, estimating and interpreting FST. *Nature Reviews Genetics*, *10*(9), pp.639-650.

Holsinger, K.E., 2006. Lecture notes in population genetics. *Storrs—Mansfield: Dept. Ecology and Evolutionary Biology, University of Connecticut. httpz/ldarwin. eeb. uconn. edu/eeb348/lecture—notes/notes. html*..

Hornik, K., 2012. The comprehensive R archive network. *Wiley Interdisciplinary Reviews: Computational Statistics*, *4*(4), pp.394-398.

Hotelling, H., 1933. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, *24*(6), p.417.

Hurlbert, S.H., 1971. The nonconcept of species diversity: a critique and alternative parameters. *Ecology*, *52*(4), pp.577-586.

Jackman, S. 2003. R for the political methodologist.*The Political Methodologist*. 11(1): 20–22.

Jin, L. and Nei, M., 1990.Limitations of the evolutionary parsimony method of phylogenetic analysis. *Molecular biology and evolution*, *7*(1), pp.82-102.

Jolliffe, I.T., 1972. Discarding variables in a principal component analysis. I: Artificial data. *Applied statistics*, pp.160-173.

Jolliffe, I.T., 1973. Discarding variables in a principal component analysis. II: Real data. *Applied statistics*, pp.21-31.

Jolliffe, I.T., 1986. Principal Components Analysis. Springer-Verlag, New York.

Jolliffe, I.T., 1989. Rotation of ill-defined principal components. *Applied Statistics*, pp.139-147.

Jombart, T., 2008.adegenet: a R package for the multivariate analysis of genetic markers. *Bioinformatics*, *24*(11), pp.1403-1405.

Jombart, T., Devillard, S. and Balloux, F., 2010. Discriminant analysis of principal components: a new method for the analysis of genetically structured populations. *BMC genetics*, *11*(1), p.94.

Jost, L.O.U., 2008. G_{ST} and its relatives do not measure differentiation. *Molecular ecology*, *17*(18), pp.4015-4026.

Jost, L.O.U., 2009. D vs. G_{ST}: Response to and. *Molecular Ecology*, *18*(10), pp.2088-2091.

Jukes, T.H. and Cantor, C.R., 1969.Evolution of protein molecules.In: Munro, H.N. (Ed.) Mammalian Protein Metabolism. pp. 21–132, New York: Academic Press.

Karp, A., Edwards, K.J., Bruford, M., Funk, S., Vosman, B., Morgante, M., Seberg, O., Kremer, A., Boursot, P., Arctander, P. and Tautz, D., 1997. Molecular technologies for biodiversity evaluation: opportunities and challenges. *Nature biotechnology*, *15*(7), pp.625-628.

Karp, A., Seberg, O.L.E. and Buiatti, M., 1996.Molecular techniques in the assessment of botanical diversity. *Annals of Botany*, *78*(2), pp.143-149.

Kimura, M., 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of molecular evolution*, *16*(2), pp.111-120.

Kimura, M., 1981. Estimation of evolutionary distances between homologous nucleotide sequences. *Proceedings of the National Academy of Sciences*, *78*(1), pp.454-458.

Lake, J.A., 1994. Reconstructing evolutionary trees from DNA and protein sequences: paralinear distances. *Proceedings of the National Academy of Sciences*, *91*(4), pp.1455-1459.

Laloë, D., Jombart, T., Dufour, A.B. and Moazami-Goudarzi, K., 2007.Consensus genetic structuring and typological value of markers using multiple co-inertia analysis. *Genetics Selection Evolution*, *39*(5), pp.545-567.

Lande, R., 1988. Genetics and demography in biological conservation. *Science (Washington), 241*(4872), pp.1455-1460.

Lande, R., 1996. Statistics and partitioning of species diversity, and similarity among multiple communities. *Oikos*, pp.5-13.

Lebart, L., 2013.Correspondence analysis. In *Data Science, Classification, and Related Methods: Proceedings of the Fifth Conference of the International Federation of Classification Societies (IFCS-96), Kobe, Japan, March 27–30, 1996* (p. 423). Springer Science & Business Media.

Leeuw, J.D., 1983. On the prehistory of correspondence analysis. *StatisticaNeerlandica*, *37*(4), pp.161-164.

Legendre, P. and Legendre, L.,1998. *Numerical Ecology.* 2nd ed. Amsterdam: Elsevier.

Leutenegger, A.L., Prum, B., Génin, E., Verny, C., Lemainque, A., Clerget-Darpoux, F. and Thompson, E.A., 2003.Estimation of the inbreeding coefficient through use of genomic data. *The American Journal of Human Genetics*, *73*(3), pp.516-523.

Lockhart, P.J., Steel, M.A., Hendy, M.D. and Penny, D., 1994. Recovering evolutionary trees under a more realistic model of sequence evolution. *Molecular biology and evolution*, *11*(4), pp.605-612.

London.

Ludwig, J.A. and Reynolds, J.F., 1988. *Statistical ecology: a primer in methods and computing* (Vol. 1). John Wiley & Sons.

Lynch, M. and Crease, T.J., 1990. The analysis of population survey data on DNA sequence variation. *Molecular biology and evolution*, *7*(4), pp.377-394.

MacQueen, J., 1967, June. Some methods for classification and analysis of multivariate observations.In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, No. 14, pp. 281-297).

Manel, S., Schwartz, M.K., Luikart, G. and Taberlet, P., 2003. Landscape genetics: combining landscape ecology and population genetics. *Trends in ecology & evolution*, *18*(4), pp.189-197.

McGuire, G., Prentice, M.J. and Wright, F., 1999.Improved error bounds for genetic distances from DNA sequences. *Biometrics*, *55*(4), pp.1064-1070.

Meirmans, P.G. and Hedrick, P.W., 2011. Assessing population structure: FST and related measures. *Molecular Ecology Resources*, *11*(1), pp.5-18.

Meirmans, P.G., 2006. Using the AMOVA framework to estimate a standardized genetic differentiation measure. *Evolution*, *60*(11), pp.2399-2402.

Meirmans, P.G., 2006. Using the AMOVA framework to estimate a standardized genetic differentiation measure. *Evolution*, *60*(11), pp.2399-2402.

Meirmans, P.G., 2012. AMOVA-based clustering of population genetic data. *Journal of Heredity*, *103*(5), pp.744-750.

Mengoni, A. and Bazzicalupo, M., 2002.The statistical treatment of data and the analysis of molecular variance (AMOVA) in molecular microbial ecology. *Annals of microbiology*, *52*(2), pp.95-102.

Mettler, L.E. and Gregg, T.G., 1969.Population genetics and evolution.*Population genetics and evolution.*

Michalakis, Y. and Excoffier, L., 1996. A generic estimation of population subdivision using distances between alleles with special reference for microsatellite loci. *Genetics*, *142*(3), pp.1061-1064.

Michaux, J.R., Magnanou, E., Paradis, E., Nieberding, C. and Libois, R., 2003. Mitochondrial phylogeography of the woodmouse (Apodemussylvaticus) in the Western Palearctic region. *Molecular Ecology*, *12*(3), pp.685-697.

Milligan, B.G., Leebens-Mack, J. and Strand, A.E., 1994. Conservation genetics: beyond the maintenance of marker diversity. *Molecular Ecology*, *3*(4), pp.423-435.

Mohammadi, S.A. and Prasanna, B.M., 2003. Analysis of genetic diversity in crop plants—salient statistical tools and considerations. *Crop science*, *43*(4), pp.1235-1248.

Moritz, C., 1994. Applications of mitochondrial DNA analysis in conservation: a critical review. *Molecular Ecology*, *3*(4), pp.401-411.

Nei, M. and Chesser, R.K., 1983.Estimation of fixation indices and gene diversities. *Annals of human genetics*, *47*(3), pp.253-259.

Nei, M., 1972.Genetic distance between populations. *The American Naturalist*, *106*(949), pp.283-292.

Nei, M., 1973. Analysis of gene diversity in subdivided populations. *Proceedings of the National Academy of Sciences*, *70*(12), pp.3321-3323.

Nei, M., 1975. Molecular population genetics and evolution. North-Holland Publishing Company.

Nei, M., 1977. F□statistics and analysis of gene diversity in subdivided populations. Annals of *human genetics*, *41*(2), pp.225-233.

Nei, M., 1978.Estimation of average heterozygosity and genetic distance from a small number of individuals. *Genetics*, *89*(3), pp.583-590.

Nei, M., 1987. *Molecular evolutionary genetics*.Columbia university press.

Nybom, H., 2004. Comparison of different nuclear DNA markers for estimating intraspecific genetic diversity in plants. *Molecular ecology*, *13*(5), pp.1143-1155.

Oksanen, J., Blanchet, F.G., Kindt, R., Legendre, P., Minchin, P.R., O'hara, R.B., Simpson, G.L., Solymos, P., Stevens, M.H.H., Wagner, H. and Oksanen, M.J., 2013. Package 'vegan'. *Community ecology package, version*, *2*(9).

Ouborg, N.J., 2009. Integrating population genetics and conservation biology in the era of genomics.*Biology Letters, 6,* pp.3-6.

Peakall, R.O.D. and Smouse, P.E., 2006. GENALEX 6: genetic analysis in Excel. Population genetic software for teaching and research. *Molecular ecology notes*, *6*(1), pp.288-295.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), pp.559-572.

Pielou, E.C., 1975. Ecological Diversity Wiley & Sons. *New York*.

Prevosti, A., 1974. La distanciagenetica entre poblaciones. *Miscellanea Alcobe, 68,* pp.109-118.

Prevosti, A., Ocana, J. and Alonso, G., 1975. Distances between populations ofDrosophilasubobscura, based on chromosome arrangement frequencies. *TAG Theoretical and Applied Genetics*, *45*(6), pp.231-241.

Pritchard, J.K., Stephens, M. and Donnelly, P., 2000.Inference of population structure using multilocus genotype data. *Genetics*, *155*(2), pp.945-959.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M.A., Bender, D., Maller, J., Sklar, P., De Bakker, P.I., Daly, M.J. and Sham, P.C., 2007. PLINK: a tool set for whole-genome association and population-based linkage analyses. *The American Journal of Human Genetics*, *81*(3), pp.559-575.

R Core Team.2017. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL: http://www.R-project.org/.

Rafalski, A., 2002. Applications of single nucleotide polymorphisms in crop genetics. *Current opinion in plant biology*, *5*(2), pp.94-100.

Raymond, M. and Rousset, F., 1995. GENEPOP (version 1.2): population genetics software for exact tests and ecumenicism. *Journal of heredity*, *86*(3), pp.248-249.

Reynolds, J., Weir, B.S. and Cockerham, C.C., 1983. Estimation of the coancestry coefficient: basis for a short-term genetic distance. *Genetics*, *105*(3), pp.767-779.

Robertson, A. and Hill, W.G., 1984. Deviations from Hardy-Weinberg proportions: sampling variances and use in estimation of inbreeding coefficients. *Genetics*, *107*(4), pp.703-718.

Rogers, J.S., 1972. Measures of genetic similarity and genetic distance. *Studies in genetics*, *7*(7213), pp.145-153.

Ryman, N. and Leimar, O., 2009. G_{ST} is still a useful measure of genetic differentiation—a comment on Jost's D. *Molecular Ecology*, *18*(10), pp.2084-2087.

Saitou, N. and Nei, M., 1987. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, *4*(4), pp.406-425.

Schonewald, C.M.C., MacBryde, S.M. and Thomas, B., 2003. *Genetics and conservationa reference for managing wild animal and plant populations* (No. 333.9516 G4).

Schonewald-Cox, C.M., Chambers, S.M., MacBryde, B. and Thomas, W.L., 1983.Genetics and conservation; a reference for managing wild animals and plant populations.

Shannon, C.E., 1948. BA mathematical theory of communication.*Bell Systems Technical Journal, 27*(3), pp.379423.

Shete, S., Tiwari, H. and Elston, R.C., 2000.On estimating the heterozygosity and polymorphism information content value. *Theoretical Population Biology*, *57*(3), pp.265-271.

Simpson, E.H., 1949. Measurement of Diversity. *Nature, 163*.

Smith, J.M., Smith, N.H., O'Rourke, M. and Spratt, B.G., 1993. How clonal are bacteria?. *Proceedings of the National Academy of Sciences*, *90*(10), pp.4384-4388.

Steemers, F.J. and Gunderson, K.L., 2007. Whole genome genotyping technologies on the BeadArray™ platform. *Biotechnology journal*, *2*(1), pp.41-49.

Stoddart, J.A. and Taylor, J.F., 1988. Genotypic diversity: estimation and prediction in samples. *Genetics*, *118*(4), pp.705-711.

Studier, J.A. and Keppler, K.J., 1988.A note on the neighbor-joining algorithm of Saitou and Nei. *Molecular biology and evolution*, *5*(6), pp.729-731.

Takezaki, N. and Nei, M., 1996.Genetic distances and reconstruction of phylogenetic trees from microsatellite DNA. *Genetics*, *144*(1), pp.389-399.

Tamura, K. and Nei, M., 1993.Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular biology and evolution*, *10*(3), pp.512-526.

Tamura, K., 1992. Estimation of the number of nucleotide substitutions when there are strong transition-transversion and G+ C-content biases. *Molecular biology and evolution*, *9*(4), pp.678-687.

Templeton, A.R., 2006. *Population genetics and microevolutionary theory*.John Wiley & Sons.

Vance, A.2009. Data analysts captivated by R's Power.*New York Times*.2009-01-06.

Vane-Wright, R.I., Humphries, C.J. and Williams, P.H., 1991. What to protect?—Systematics and the agony of choice. *Biological conservation*, *55*(3), pp.235-254.

Verity, R. and Nichols, R.A., 2014. What is genetic differentiation, and how should we measure it—G_{ST}, D, neither or both?. *Molecular ecology*, *23*(17), pp.4216-4225.

Vignal, A., Milan, D., SanCristobal, M. and Eggen, A., 2002.A review on SNP and other types of molecular markers and their use in animal genetics. *Genetics Selection Evolution*, *34*(3), pp.275-306.

Vogl, C., Karhu, A., Moran, G. and Savolainen, O., 2002. High resolution analysis of mating systems: inbreeding in natural populations of Pinusradiata. *Journal of Evolutionary Biology*, *15*(3), pp.433-439.

Wang, J., 2011a. A new likelihood estimator and its comparison with moment estimators of individual genome-wide diversity. *Heredity*, *107*(5), pp.433-443.

Wang, J., 2011b. Unbiased relatedness estimation in structured populations. *Genetics*, *187*(3), pp.887-901.

Weir, B.S. 1996.Intraspecific differentiation. p. 385–403. In:Hillis, C.M. and Mable, B.K., (Eds.) Molecular systematics.2nd edition, Sinauer Associates, Sunderland, MA.

1996. Molecular Systematics, —David M. *Syst. Biol*, *45*(4), pp.607-609.

Weir, B.S. and Cockerham, C.C., 1984.Estimating F-statistics for the analysis of population structure. *evolution*, pp.1358-1370.

Weir, B.S., Cardon, L.R., Anderson, A.D., Nielsen, D.M. and Hill, W.G., 2005. Measures of human population structure show heterogeneity among genomic regions. *Genome research*, *15*(11), pp.1468-1476.

Whitlock, M.C., 2011. $G'_{ST}$and D do not replace F_{ST}. *Molecular Ecology*, *20*(6), pp.1083-1091.

Wright, S., 1921a.Systems of mating. I. The biometric relations between parent and offspring. *Genetics*, *6*(2), p.111.

Wright, S., 1921b.Systems of mating. II. The effects of inbreeding on the genetic composition of a population. *Genetics*, *6*(2), p.124.

Wright, S., 1922.Coefficients of inbreeding and relationship. *The American Naturalist*, *56*(645), pp.330-338.

Zhang, F. and Ge, S., 2001.Data analysis in population genetics. I .analysis of RAPD data with AMOVA. *Chinese Biodiversity*, *10*(4), pp.438-444.

Application of R Software in Analysis of Genetic Diversity and Population Genetics

By:

Mehdi Zahravi (Ph.D.)