例題 6.1

1. R パッケージ ape、FinePop2、sf、tibble、RColorBrewer をインストールせよ。
2. Figshare（https://doi.org/10.25387/g3.14813490）に公開されているヒトのマイクロサテライト遺伝子型（377 遺伝子座）および地図データと R script をダウンロードし、図 6.4 などが作図されることを確認せよ。
3. 図 6.4c では、集団対 FST の距離行列（`pair.fst` オブジェクト）から系統樹で集団構造を描いている。ここでは MDS で描画し、ヒトの集団構造について考察せよ。

1 の解答例

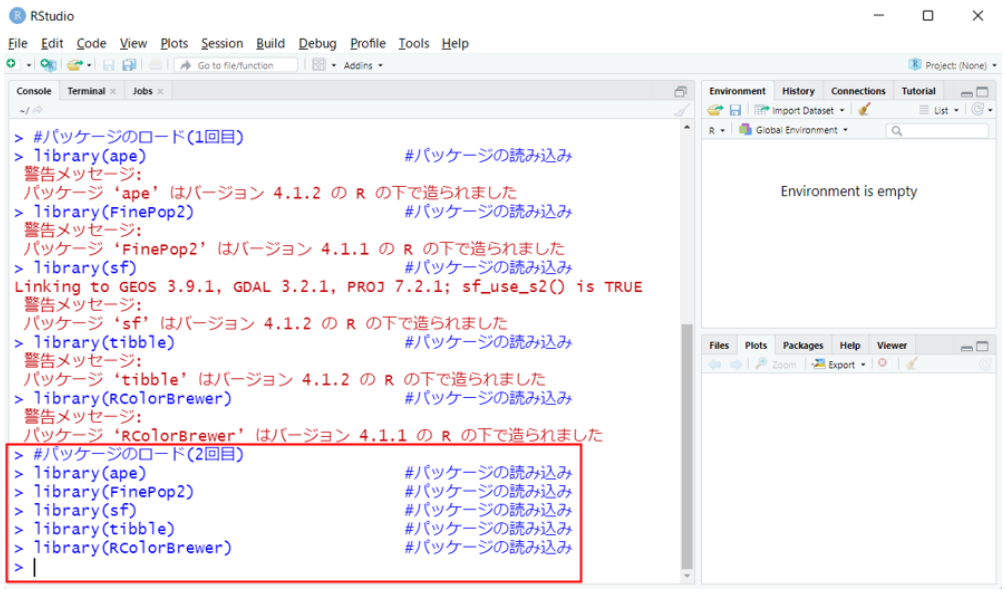
パッケージのインストールは、以下のコマンドで実行可能です。実際の作業は、ウェブ資料中のコマンドをコピペ実行してください。

```
install.packages("ape")
install.packages("FinePop2")
install.packages("sf")
install.packages("tibble")
install.packages("RColorBrewer")
```

念のため、以下のコマンドを実行してみてください。エラーメッセージが出なければ、パッケージが無事ロードできている（インストールに成功している）と判断できます。

```
library(ape)
library(FinePop2)
library(sf)
library(tibble)
library(RColorBrewer)
```

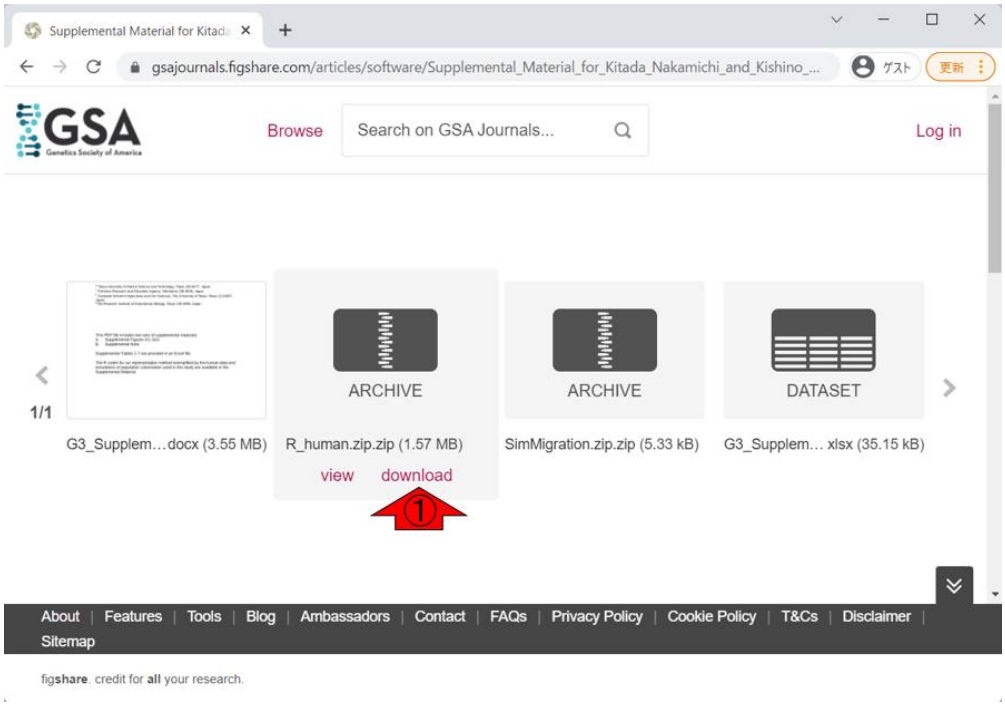
ウェブ資料中のコマンドをコピペ実行し、以下のような感じに見えていればインストール成功です。



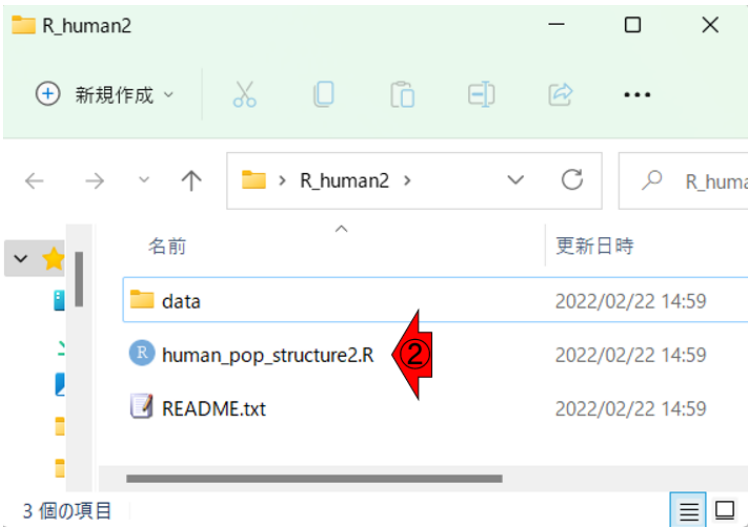
1 回目のロードで警告メッセージが出ていても、2 回目のロード実行結果が赤枠のように見えていれば OK です。

2 の解答例

figshare (https://doi.org/10.25387/g3.14813490) にアクセスすると以下のように見えます。

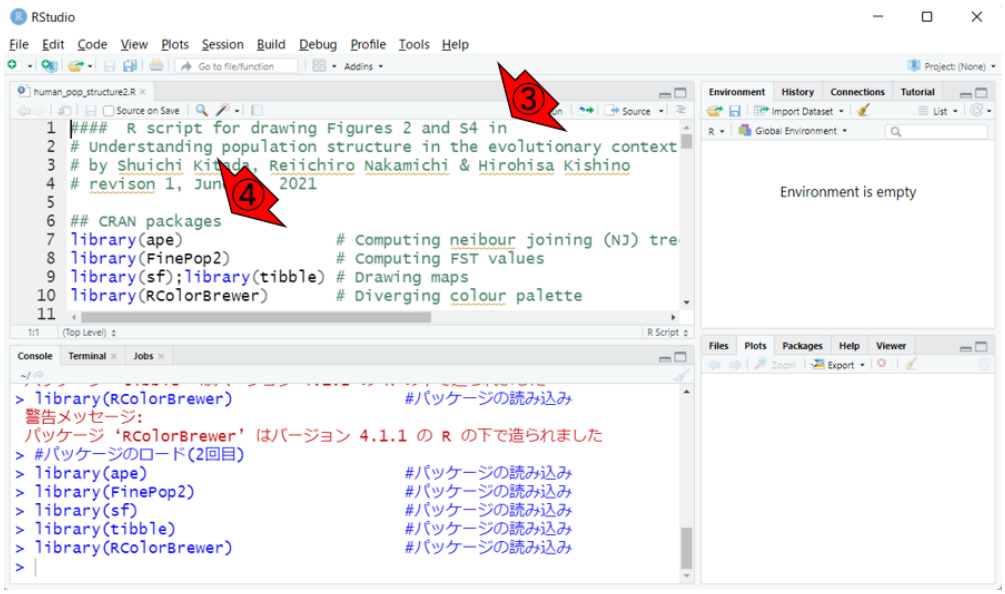


「ヒトのマイクロサテライト遺伝子型（377 遺伝子座）および地図データと R script」に相当するファイルは R_human.zip.zip です。①をクリックしてダウンロードします。



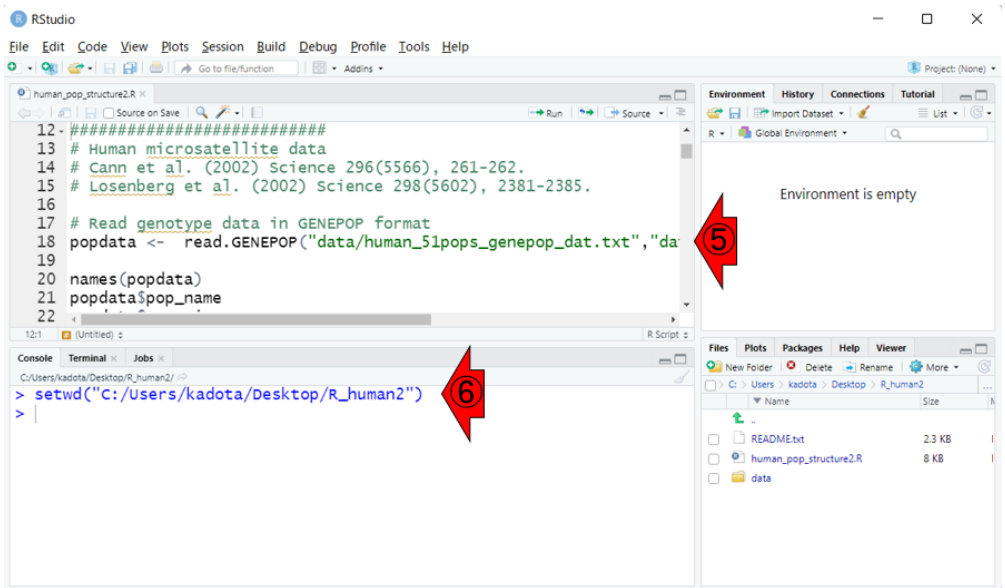
Zip 圧縮ファイルを解凍すると R_human2 というフォルダが得られます。この解答例作成環境では、「デスクトップ」上に R_human2 フォルダが存在します。

RStudio 上で R_human2 フォルダ内に存在する②human_pop_structure2.R を開くと、（環境によって若干異なりますが概ね）以下のように見えると思います。



③のあたりに、この R スクリプトファイルが原著論文の Fig. 2 と S4 を作成するためのものだと書かれていることがわかります。この Fig. 2 が作成したい図 6.4 に対応します。④で見えているのは、さきほど実行したパッケージ（ライブラリとも表現されます）のロードと同じですので、やる必要はありません。

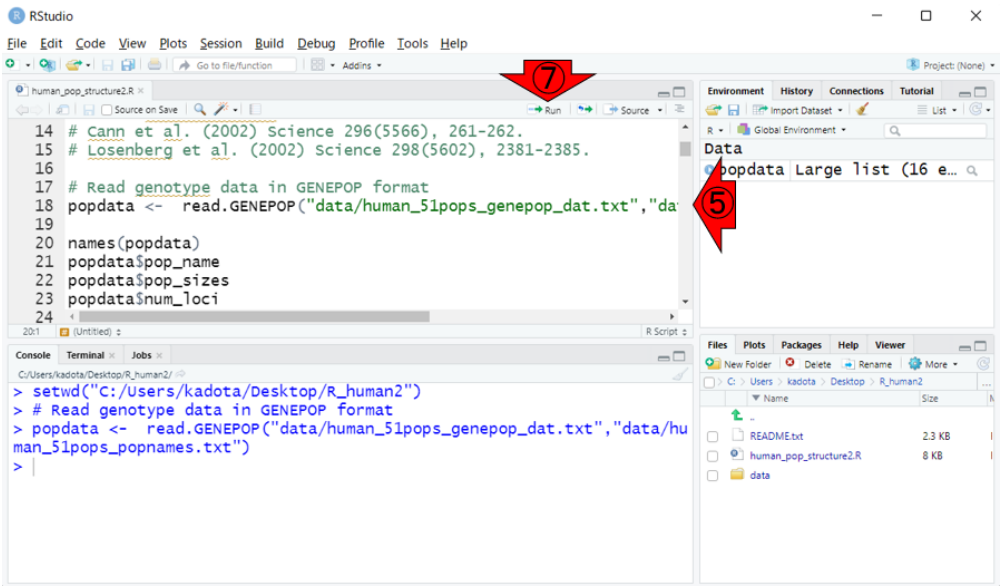
以下のスクショ中の⑤は、今開いている human_pop_structure2.R の 18 行目です。ここでファイルの読み込みを行っています。



この「data/ human_51pops_genepop_dat.txt」を見た段階で、R_human2 フォルダを作業ディレクトリにせねばならないことがわかります。⑥のような感じで作業デ

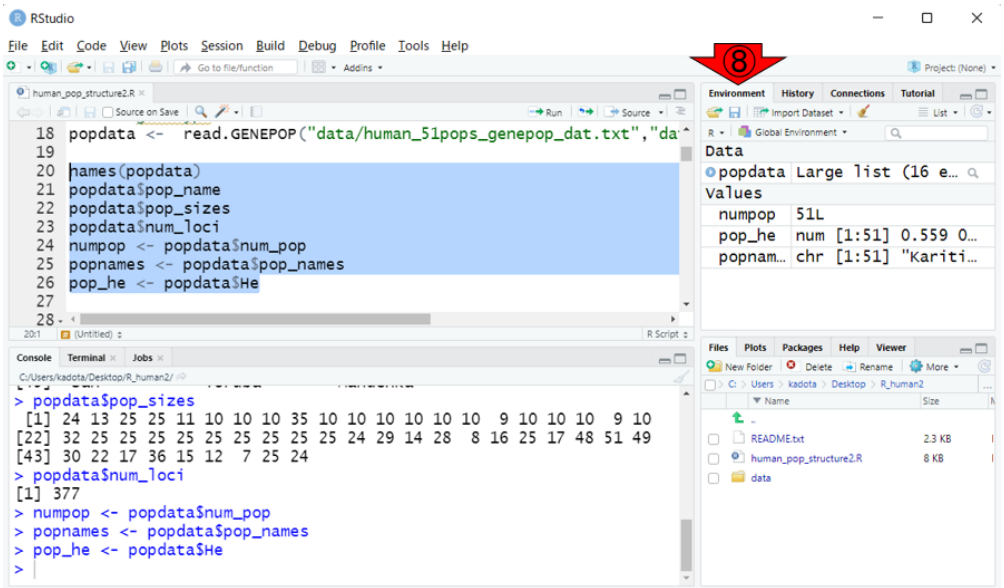
ィレクトリの変更を行います。

以下のスクショは、⑤の 18 行目のところにカーソルを移動させて、⑦実行した結果です。



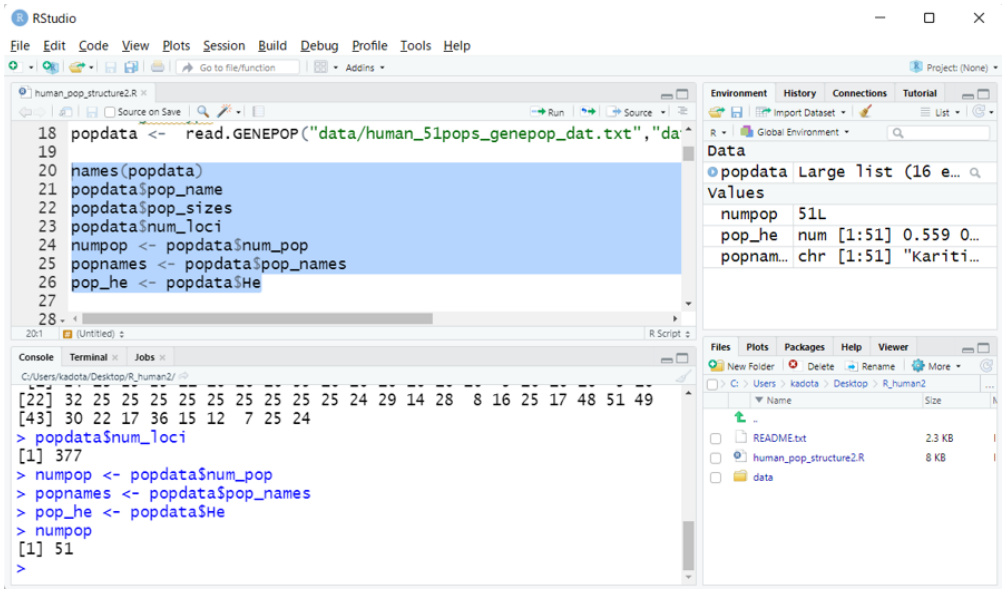
左下の Console 画面上で特に何もエラーメッセージが出ていないので、入力ファイルの読み込みが無事終わり、その中身が popdata というオブジェクトに格納されたと解釈します。

以降も基本的にコピペ実行していくだけです。例えば以下は、20-26 行目を反転させて、一気に実行した結果のスクショです。

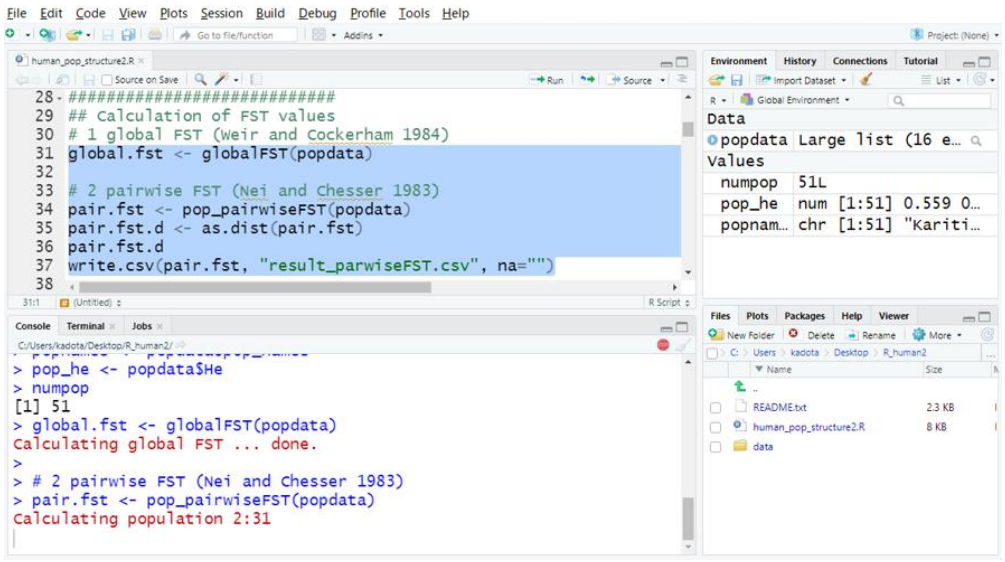


20-23 行目までは、基本的に中身の一部を表示させて確認しているだけですので、実行しなくても特段問題ありません。ただし、24-26 行目はそれぞれ numpop、

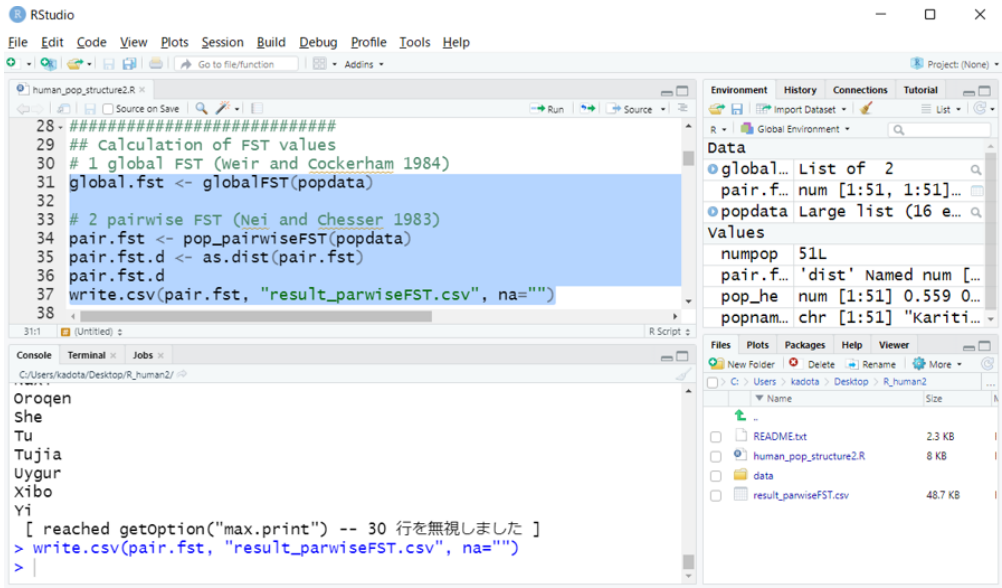
popnames、pop_he という新たなオブジェクトを作成するコマンドなので必ず実行せねばなりません。それらの中身は、右上のほうの⑧Environment というタブ内でも大まかに概要を知ることができます。例えば実質的に popdata$numpop のコピーである numpop の中身は、51 という数値です。また、pop_he は 51 個の要素からなる数値ベクトルだと判断します。スクリプトファイル（human_pop_structure2.R）中に記載されていなくても、疑問に思ったオブジェクトの中身や確認したい事柄があれば、随時確認しながら進めていくとよいと思います。例えば以下のスクショは、numpop の中身を確認しているだけです。



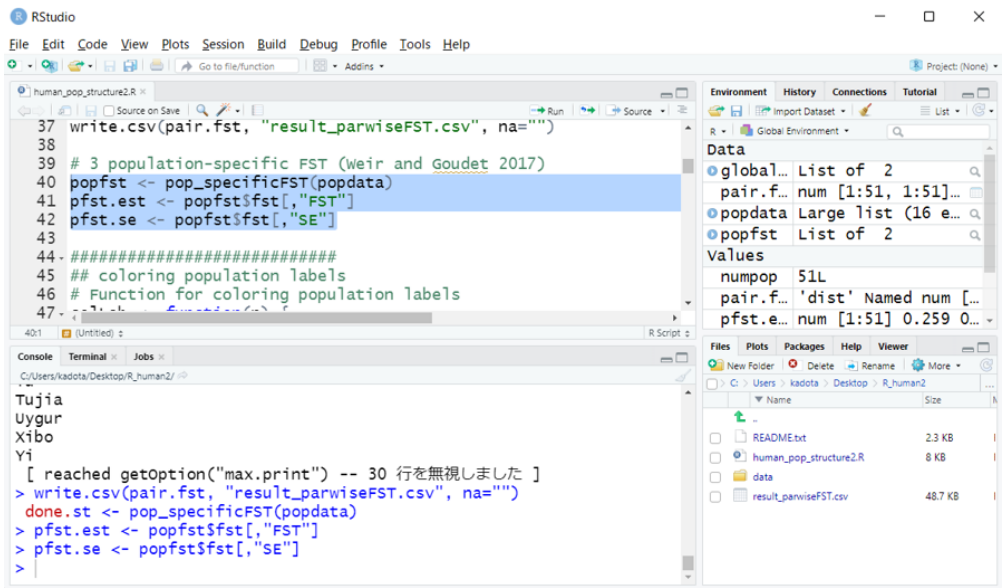
31-37 行目が F_{ST} の計算部分です（数分で終わります）。以下は計算の途中経過のスクショです。



以下のスクショは、37 行目の集団対 F_{ST} の計算結果を CSV 形式ファイル（result_parwiseFST.csv）で作業ディレクトリに保存する作業までのものです。

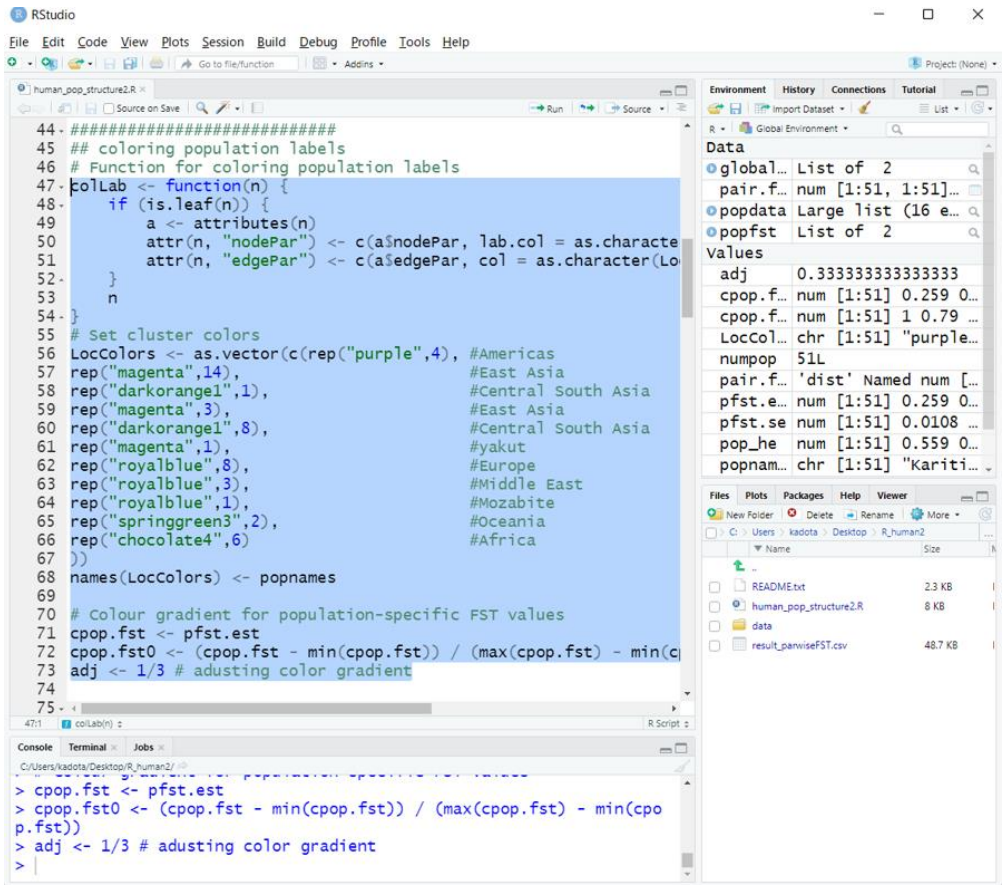


39 行目のコメントでもわかりますが、40-42 行目が集団固有 F_{ST} 計算部分です。右下の Console 画面において、done.の赤字が変なところに入っているように見えますが、エラーが出ているわけではありませんので気にしなくてよいです。

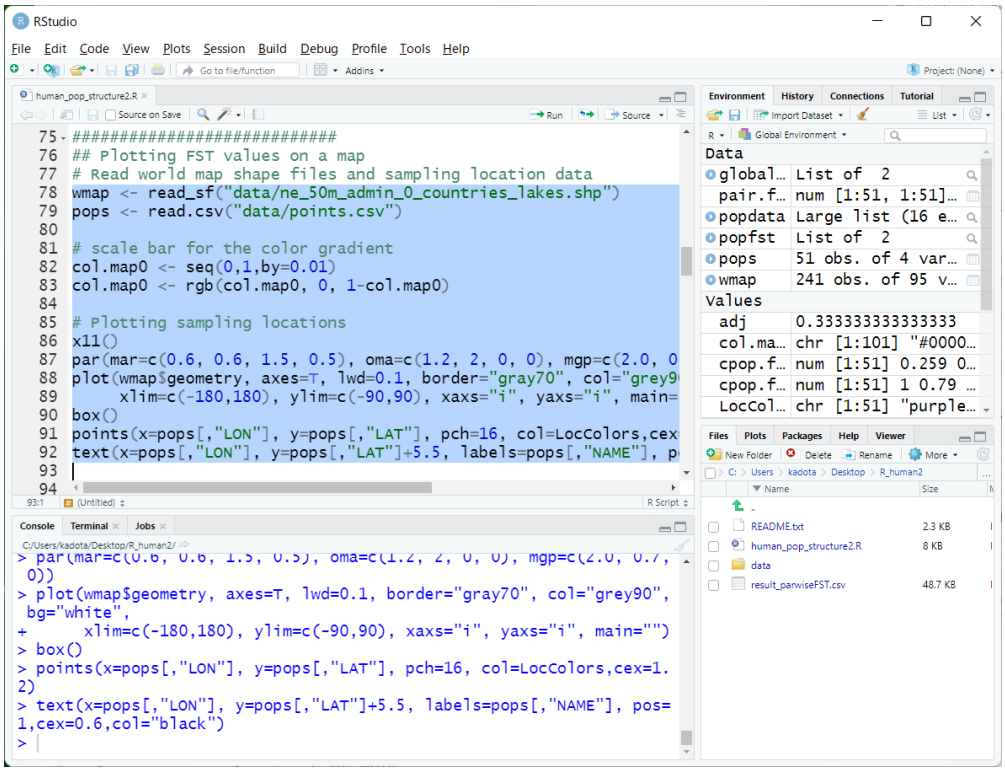


45-46 行目のコメントでもなんとなくわかりますが、47-54 行目で colLab という名前の関数を自作しています。56-68 行目では、サンプリング場所ごとにどの色を割り振るかを指定しています。71-73 行目では、指定した色の濃さを集団固有 F_{ST} 値によって変えているのだと解釈すればよいです。例えば 72 行目の数式は分母が「max - min」、分子が「当該 F_{ST} 値 - min」のようになっていますが、こ

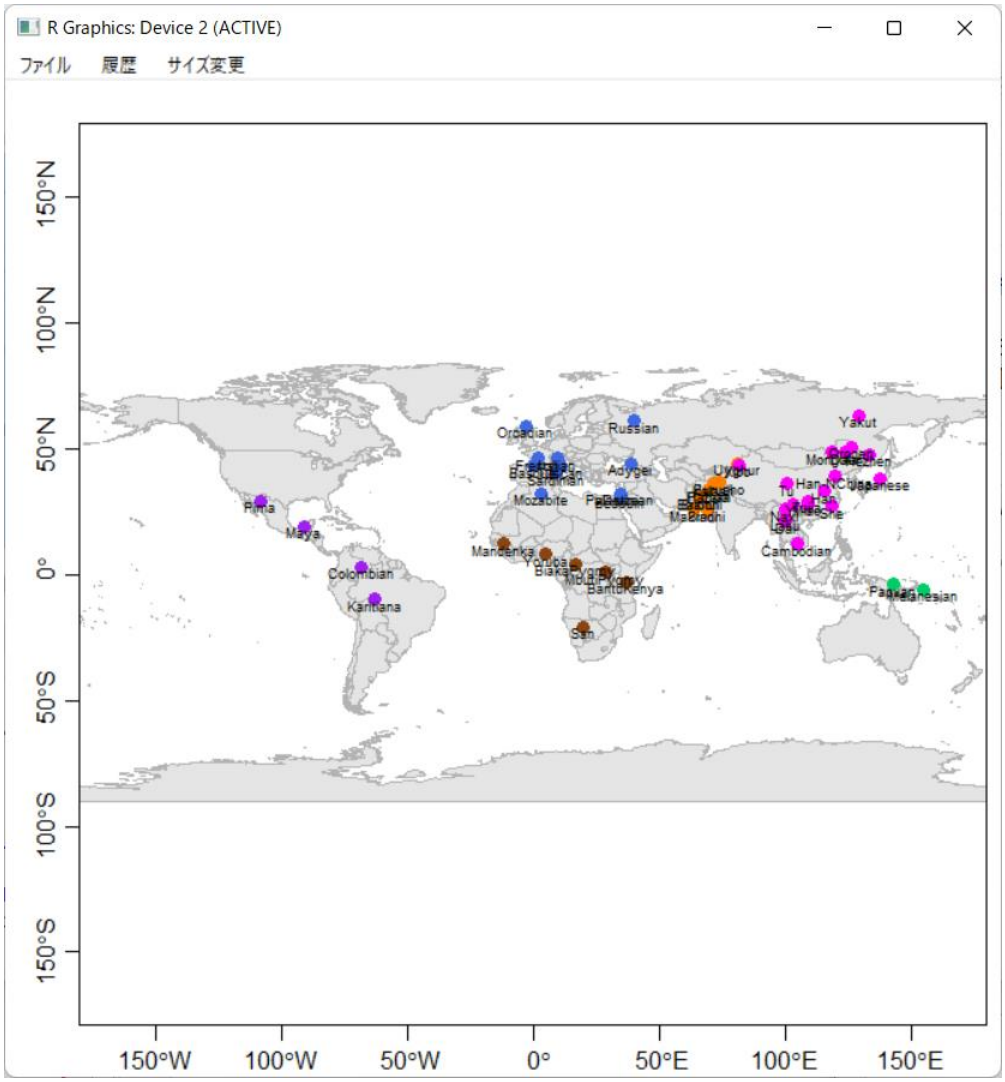
れは値を 0-100%の範囲になるように正規化したい場合に用いる式です。



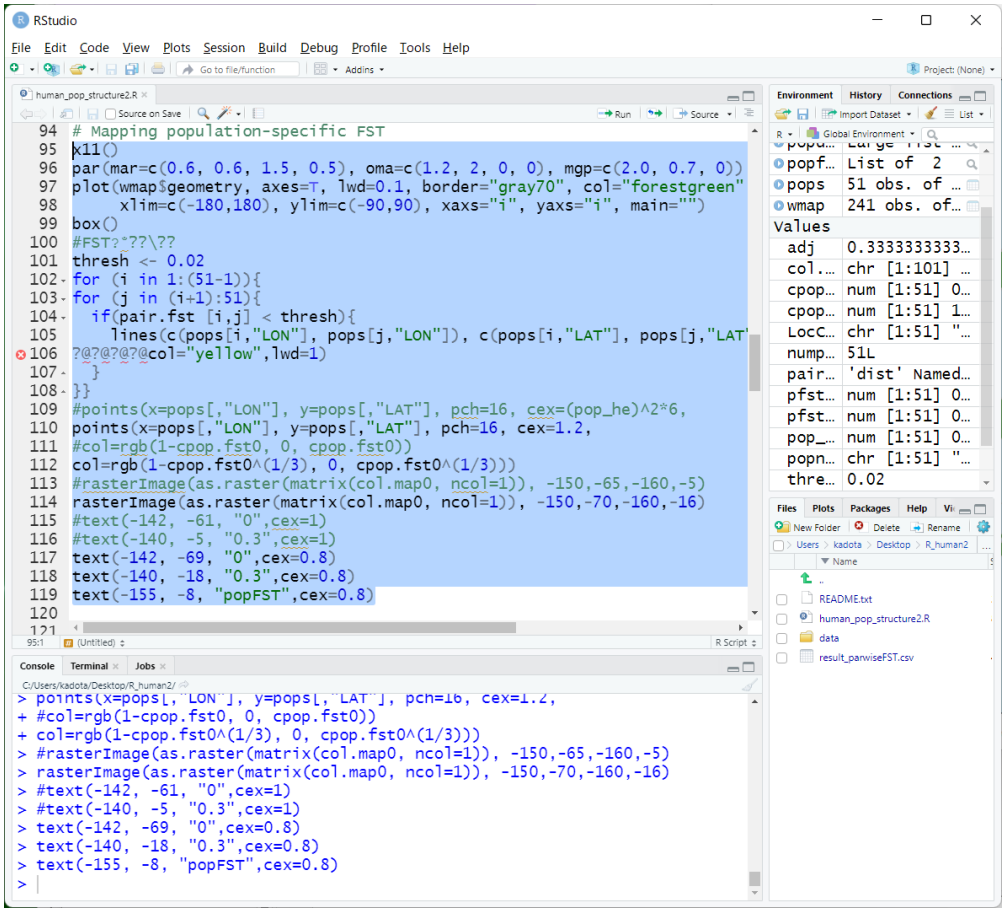
76 行目では、「世界地図上に F_{ST} 値をプロットする」と書いています。このプロットを行うためには、当然ながら世界地図の座標データと、どこでサンプリングしたか（世界地図上のどこにプロットするか）という情報を用意しておかねばなりません。77 行目で shape file と書かれていて面食らうかもしれませんが、シンプルに世界地図の形（shape）の情報を保持したファイルなのだろうと読み解けばよいです。世界地図の座標データを読み込んでいるのが 78 行目、サンプリング地点情報を読み込んでいるのが 79 行目です。78 行目で読み込んでいる入力ファイルの拡張子が.shp となっていたり、読み込むために用いている read.sf という名前の関数に戸惑うかもしれませんが、これも「特有の形式を読み込むための専用の関数がキチンと用意されているのだろう」程度の理解で実用上は問題ありません。プロットのメインは 88-89 行目の plot 関数になります。87 行目の par 関数で指定しているのは、マージン幅などのプロットに関するパラメータ（parameters）情報です。それ以外についても詳細は述べませんが、いろいろと少しづつ数値を変更して見栄えの違いを経験的に学習していくとよいでしょう。



92 行目までのスクリプトの実行が終わると、以下に示すような GUI が起動して世界地図が表示されます。同時に、(56-67 行目で指定した色で) カラフルに色分けされたサンプリング地点の情報や、その地点名の情報が黒字で示されていることがわかります。上記スクリプトと実際の描画内容を見比べることで、例えば 91 行目の points 関数が「サンプリング地点を塗りつぶした○で示す」ことに、そして 92 行目の text 関数が「黒字で書かれたサンプリング地点名情報の記載」に相当することなどが分かるようになってきます。



94 行目では、「集団固有 F_{ST} のマップ」と書かれています。96-119 行目が実際のコマンドです。100 行目で FST 以降の文字が文字化けしています。文字コードを変更すれば見られるようになりますが、これは#から始まるコメント行なので特に気にする必要はありません。明らかにおかしくなっているのは、106 行目です。これは文字化けになっている箇所を探すというよりは、RStudio が文法上の誤り箇所を自動的に示してくれる機能を頼りに見つけました。具体的には「106 の左側に小さく赤字で×となっている」箇所を探すということになります。ここは明らかにコード内部におかしな文字化けが含まれています。ただし、**環境によってはこのようなエラーが見えないこともありますので、上記エラーが見られなければ特に気にする必要はありません**。これは以降のエラーについても同様です。



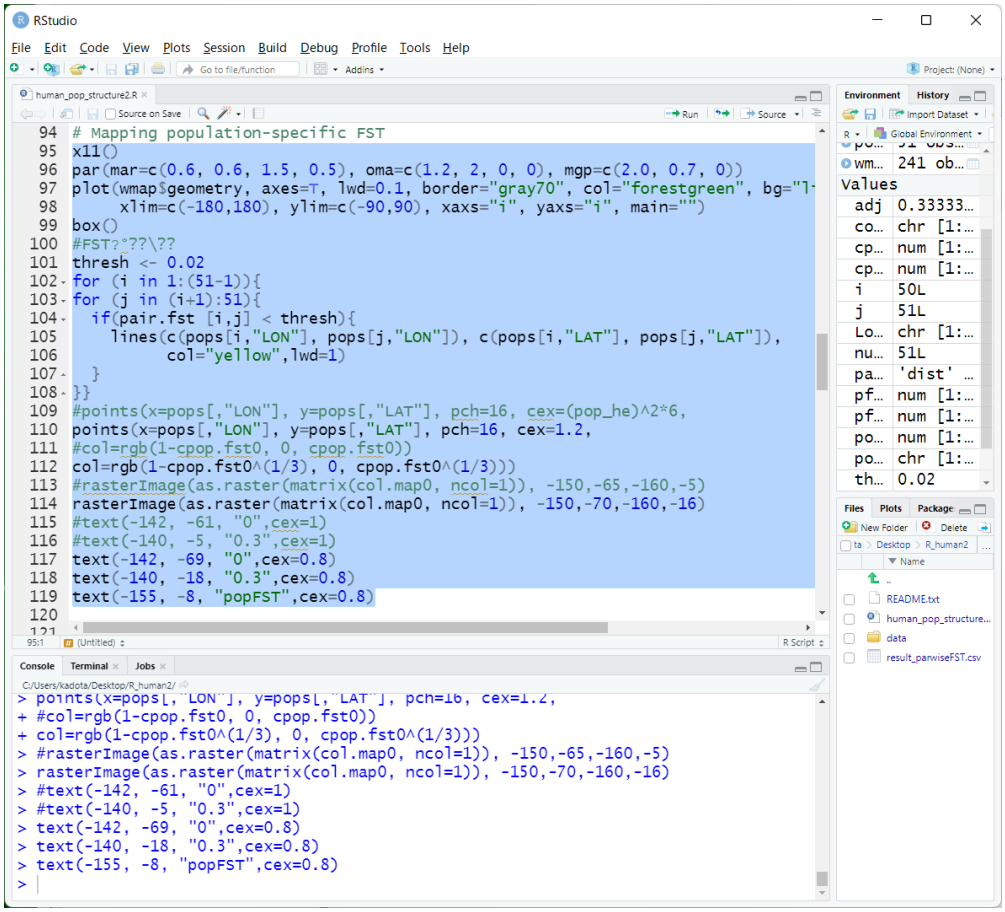
当該箇所周辺のみを示したのが以下のスクショです。よく見ると、実は 106 行目のコマンドは 105 行目の続きとして書かれているものであることが感覚的にわかります。もしかしたら「スペースキー」が半角ではなく全角になってしまっていたためにこのように見えているのかもしれません。いずれにせよ文法的におかしいと指摘されているので、ここでは文字化けしている部分を半角スペースに変更しておきます。

```
100 #FST?*??\??
101 thresh <- 0.02
102 for (i in 1:(51-1)){
103 for (j in (i+1):51){
104   if(pair.fst [i,j] < thresh){
105     lines(c(pops[i,"LON"], pops[j,"LON"]), c(pops[i,"LAT"], pops[j,"LAT"]),
106 ?@?@?@?@col="yellow",lwd=1)
107   }
108 }}
109 #points(x=pops[,"LON"], y=pops[,"LAT"], pch=16, cex=(pop_he)^2*6,
110 points(x=pops[,"LON"], y=pops[,"LAT"], pch=16, cex=1.2,
111 #col=rgb(1-cpop.fst0, 0, cpop.fst0))
112 col=rgb(1-cpop.fst0^(1/3), 0, cpop.fst0^(1/3)))
```

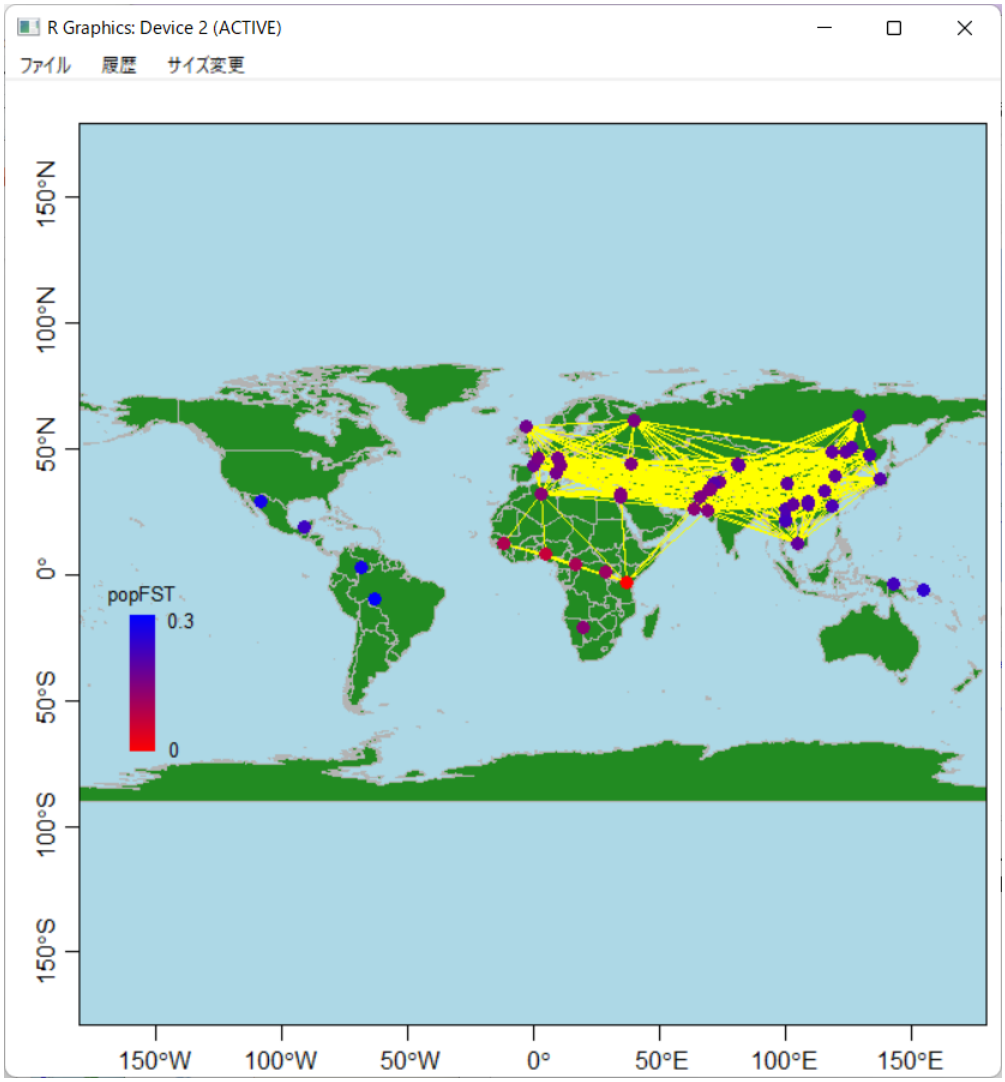
半角スペースに置き換えたものが以下のスクショです。無事 106 行目の文法エラーが消滅していることがわかります。

```
#FST?*??\??
thresh <- 0.02
for (i in 1:(51-1)){
for (j in (i+1):51){
  if(pair.fst [i,j] < thresh){
    lines(c(pops[i,"LON"], pops[j,"LON"]), c(pops[i,"LAT"], pops[j,"LAT"]),
          col="yellow",lwd=1)
  }
}}
#points(x=pops[,"LON"], y=pops[,"LAT"], pch=16, cex=(pop_he)^2*6,
points(x=pops[,"LON"], y=pops[,"LAT"], pch=16, cex=1.2,
#col=rgb(1-cpop.fst0, 0, cpop.fst0))
col=rgb(1-cpop.fst0^(1/3), 0, cpop.fst0^(1/3)))
```

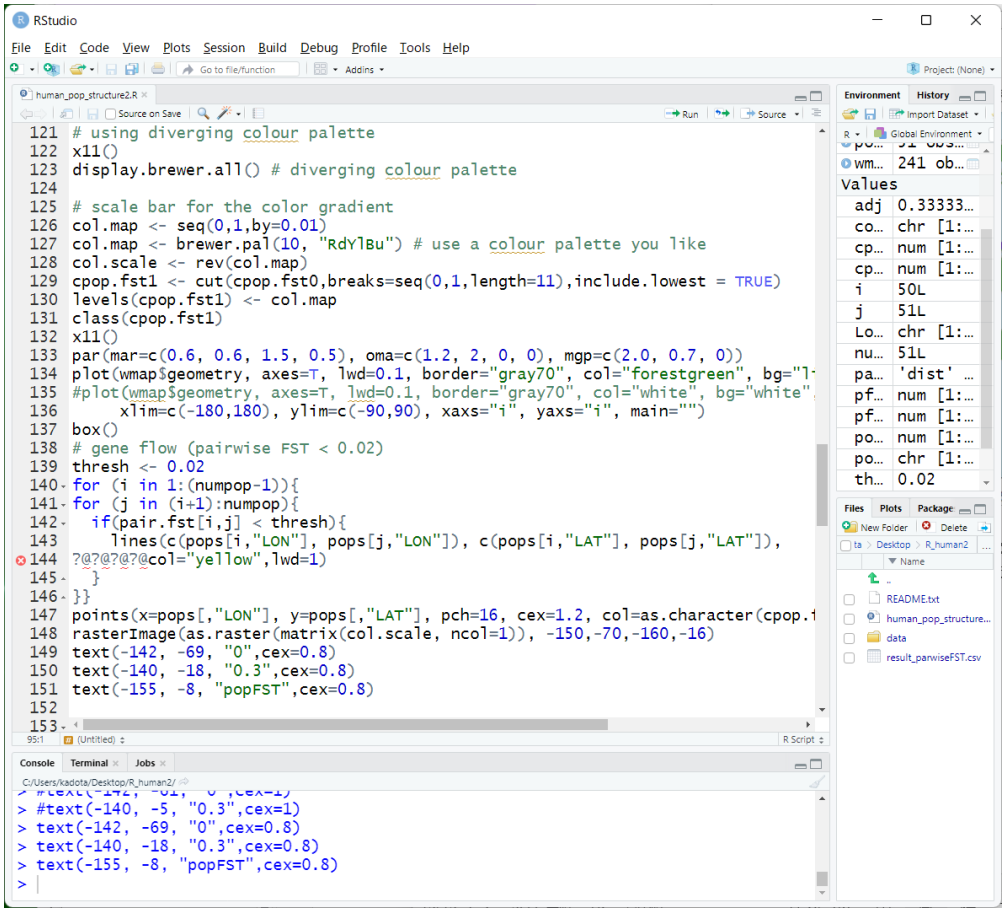
気を取り直して、改めて 95-119 行目のスクリプトを実行します。



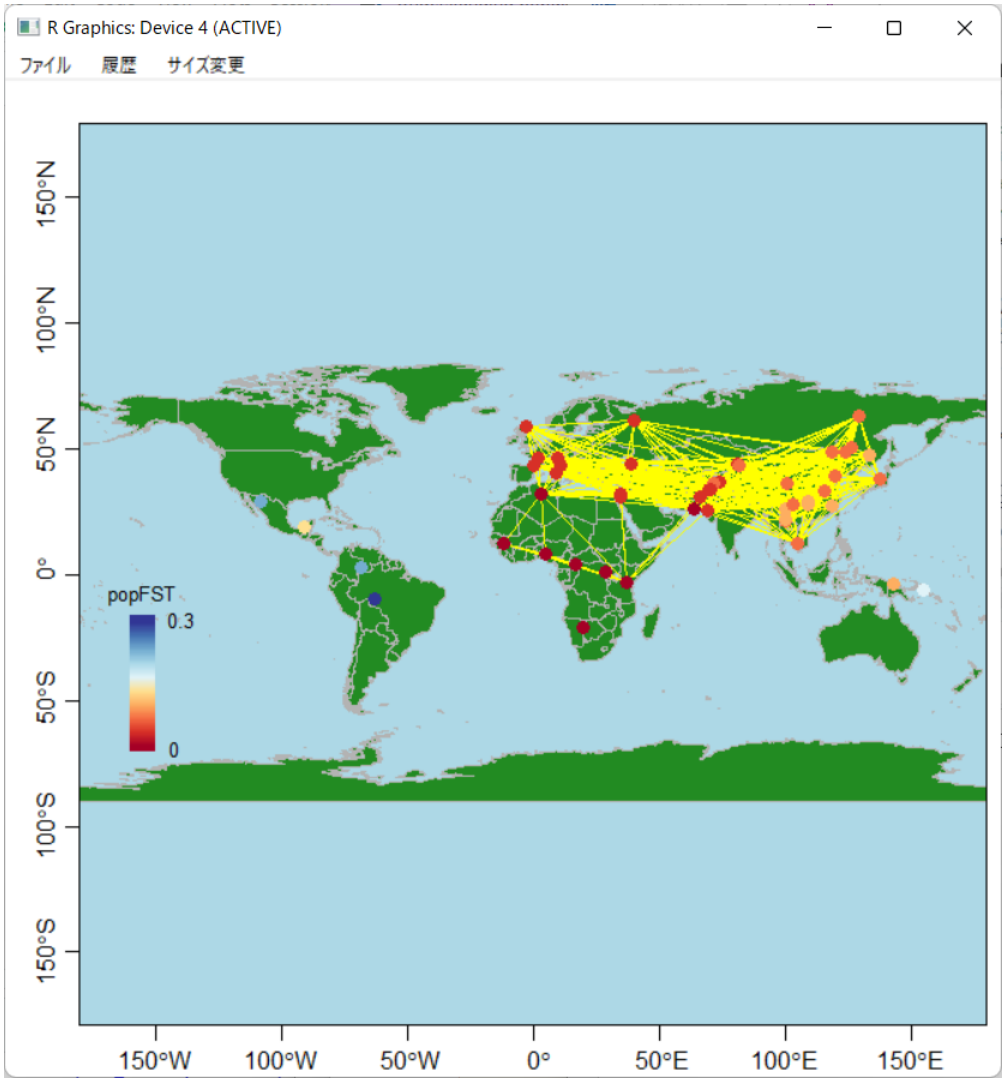
119 行目まで実行し終えると同時に、以下のような図が得られます。これは原著論文の Figure 2A と同じであり、その白黒版が図 6.4a という関係性になります。ここでは成功した結果しか示しませんが、仮に 106 行目の文字化けが含まれたままで実行すると、サンプリング地点間を結ぶ多数の黄色の線が消えた状態になります。**ここまでで図 6.4a の確認まで完了した**ことになります。



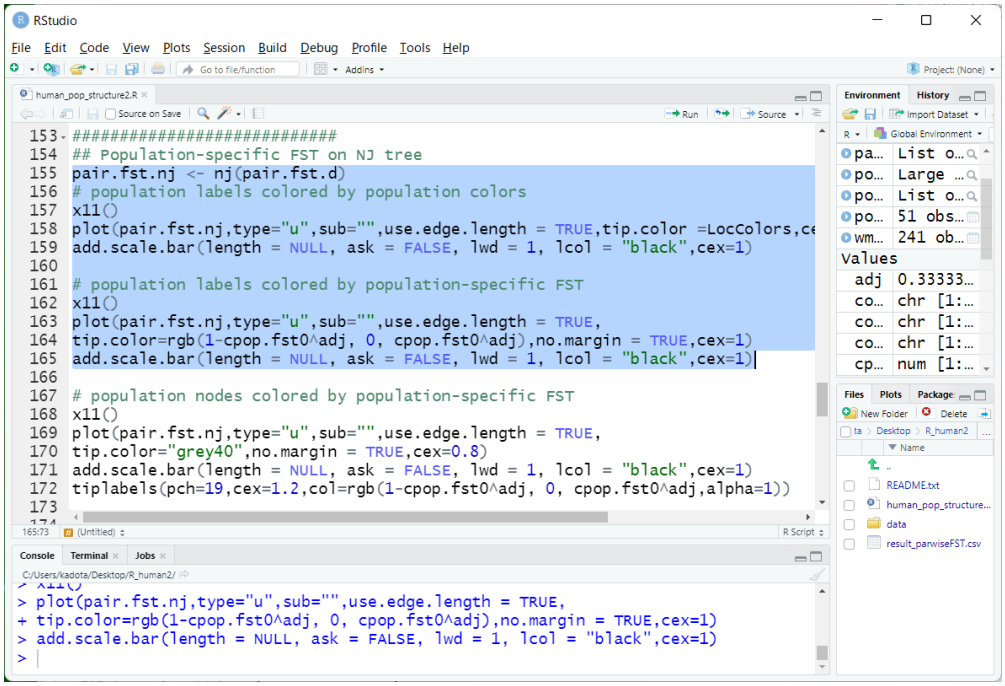
以下のスクショは、121-151 行目までのスクリプトです。ここでも先ほど述べたことと同じ文法上のエラーが 144 行目に出現していることがわかります。



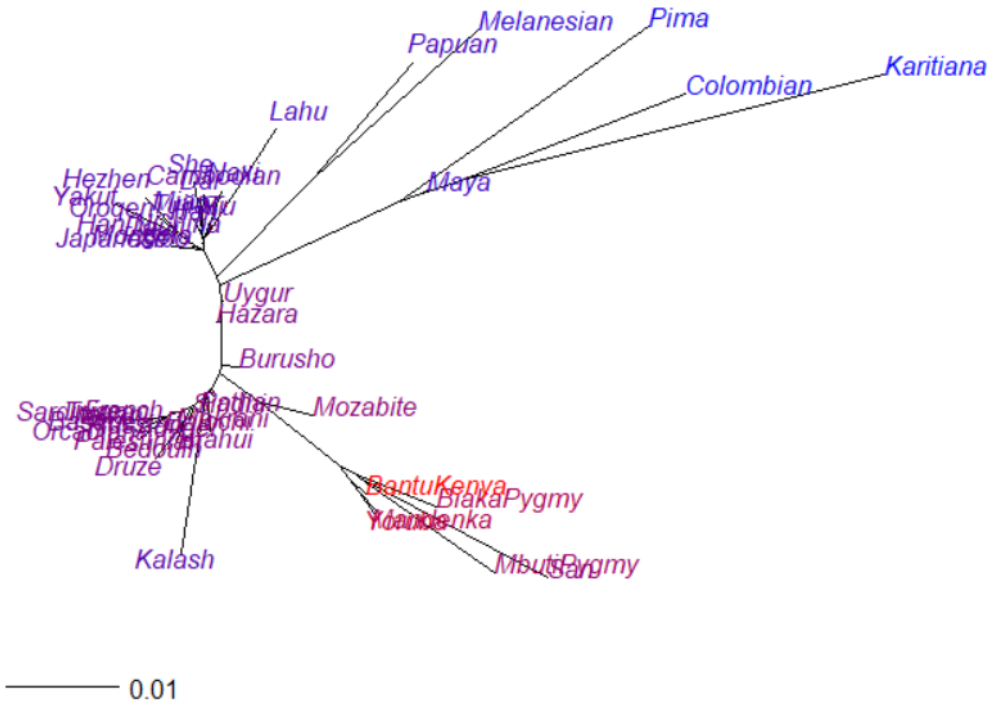
144 行目の修正を施したものが下記のスクショです。これは、さきほどの図と色合いが若干異なるもので、原著論文の Figure S4 を作成するスクリプトになります。**ここまでで、原著論文の Figure 2A の図の再現に成功したことになります。これは図 6.4a のカラー版に相当するものだから**です。



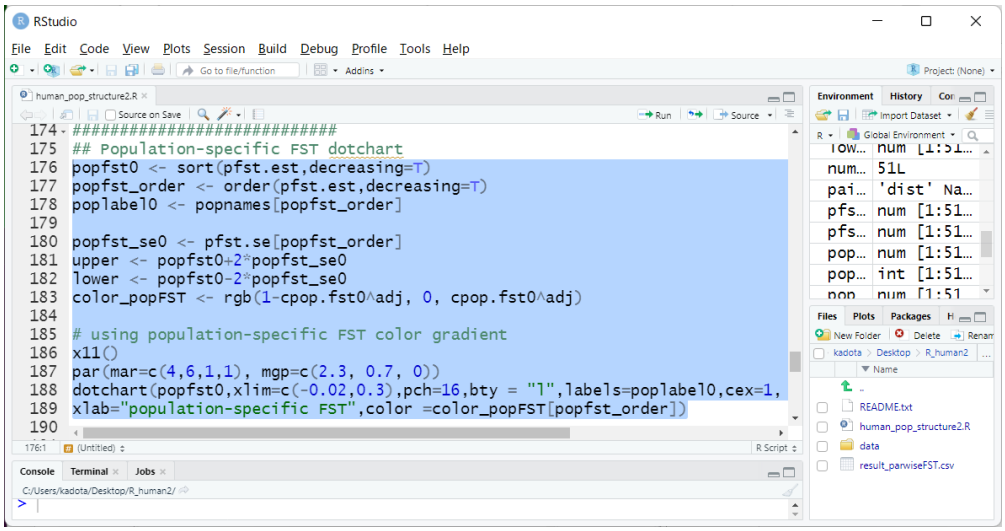
次に、図 6.4c の再現を行います。以下のスクショの 155-165 行目を実行します。これは、155 行目で NJ 樹を構築し、157-159 行目で 1 つめのプロットを、そして 162-165 行目で 2 つめのプロットを作成しています。尚、画面上で見えている 168-172 行目は、見せ方の違いだけです。ここでは示しませんが、実用上はこのような様々な見せ方の図を作成します。そして、いくつかの候補の中から最終的に論文の図としてどれを用いるのが適切かを判断します。



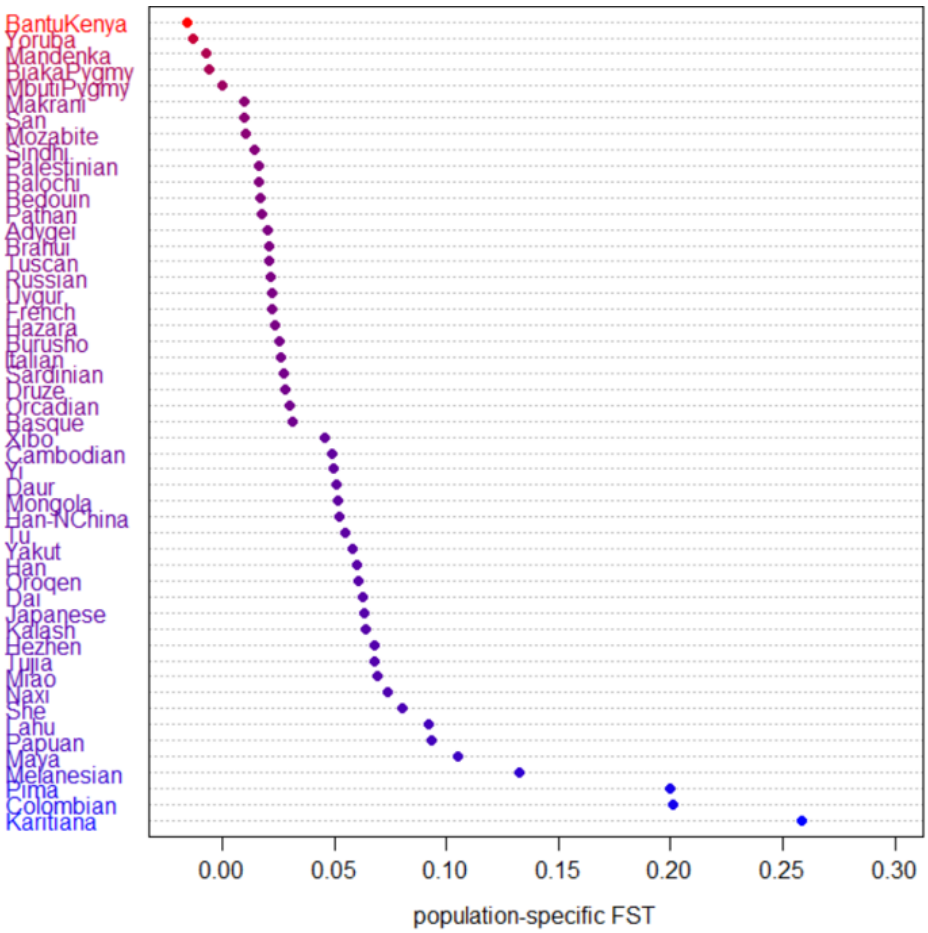
以下は、162-165 行目の 2 つめのプロットの結果です。**これは原著論文の Figure 2C と同じであり、その白黒版が図 6.4c** になります。



最後に図 6.4b の描画ですが、これはカラー版である原著論文の Figure 2B と同じです。まずは 176-189 行目を実行します。

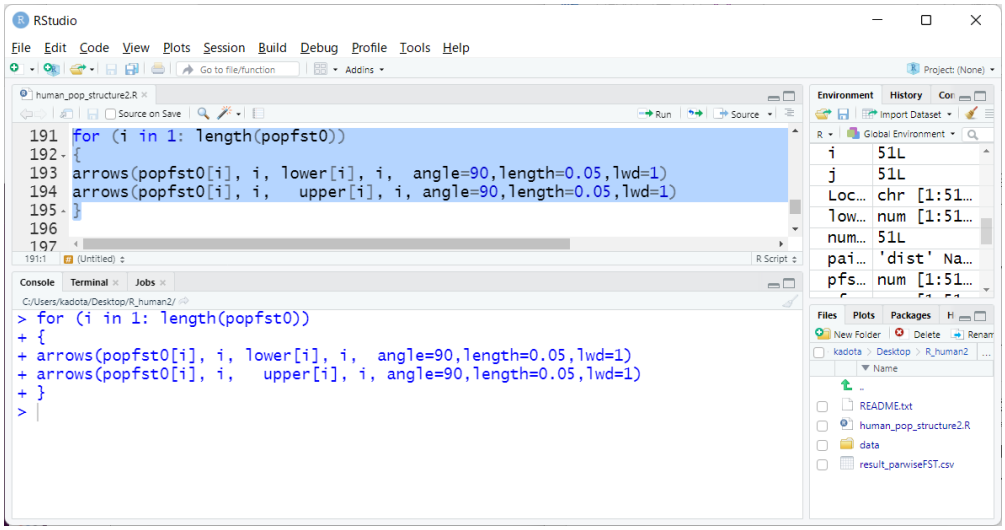


実行結果として、以下の図が得られます。

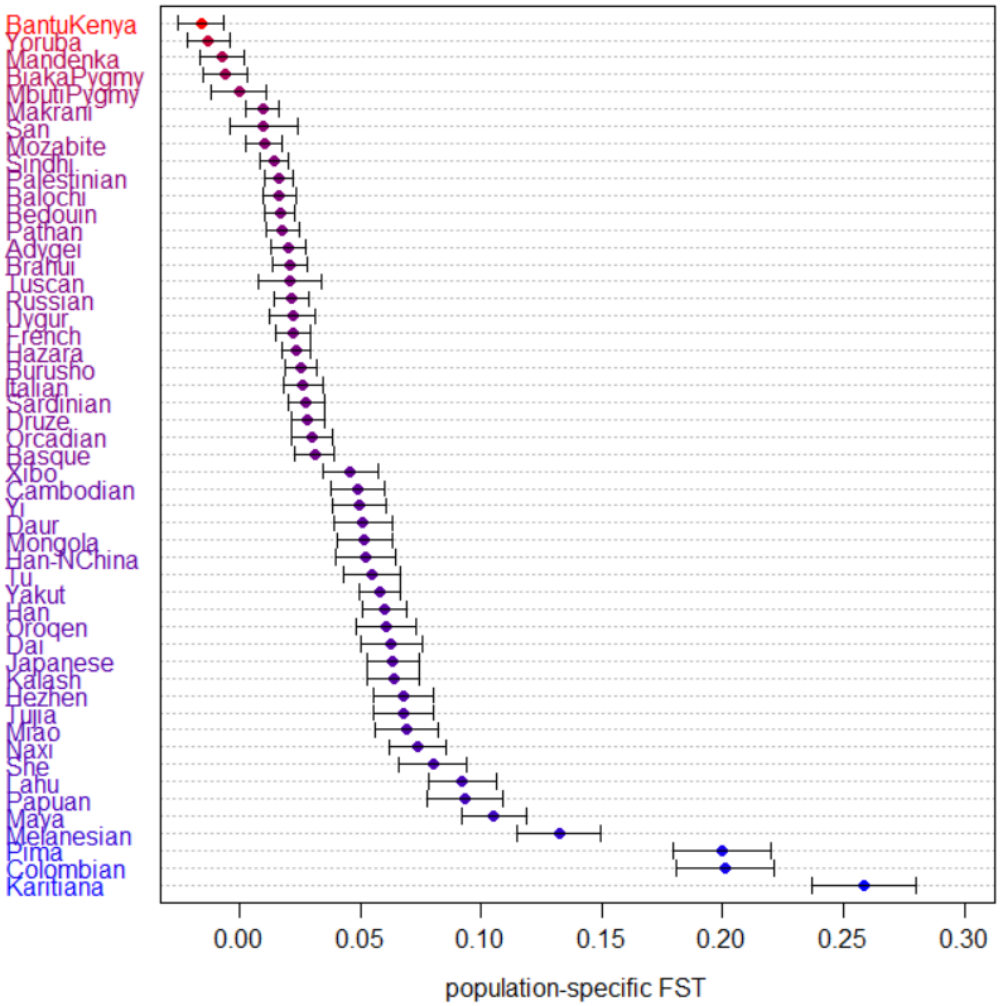


R スクリプトの 176-178 行目では、x 軸の値に相当する集団固有 F_{ST} で、y 軸に相当する地点名をソートして表示するための情報を得ています。176 行目の sort 関数実行時に decreasing=T オプションを付けているのは、値を降順（大きい値 → 小さい値）にソートせよという指令です。この図は y 軸上で下から上にプロットしているので、このオプションをつけています。178 行目の実行結果までで得られた `poplabel0` オブジェクトを表示させてみると、1 番目の要素が" Karitiana"になっている事実からも納得できると思います。また、y 軸の文字は一番下の”Karitiana”が青で、一番上の"BantuKenya"が赤っぽく見えていますが、これは 180-183 行目のところで F_{ST} 値の値に応じて表示色を変更させるべく、色を RGB で定義しています。rgb 関数で指定する 1 番目の引数（`1-cpop.fst0^adj`）が赤に、2 番目の引数（常に 0）が緑に、そして 3 番目の引数（`cpop.fst0^adj`）が青に相当します。実際のプロットが下から上に向かって青から赤に変遷していることからも想像できますが、rgb 関数で 2 番目の引数の値が固定であることは至極妥当です。186-189 行目がプロットの本番になります。

図 6.4b では、標準誤差（SE）の 2 倍の範囲を左右両方向に向かって矢印で示しています。この矢印の先端部分（終点部分）の値を定義しているのが、181-182 行目の実行結果として得られた `upper` と `lower` オブジェクトです。これらの値を得るために、180 行目で得た標準誤差（SE）の値を格納した `popfst_se0` を利用しています。矢印の起点部分の値は、176 行目で得た「降順にソートされた集団固有 F_{ST} 情報を含む `popfst0` オブジェクト」になります。これらの必要な情報を踏まえると、191-195 行目のコマンドの意味がよくわかると思います。

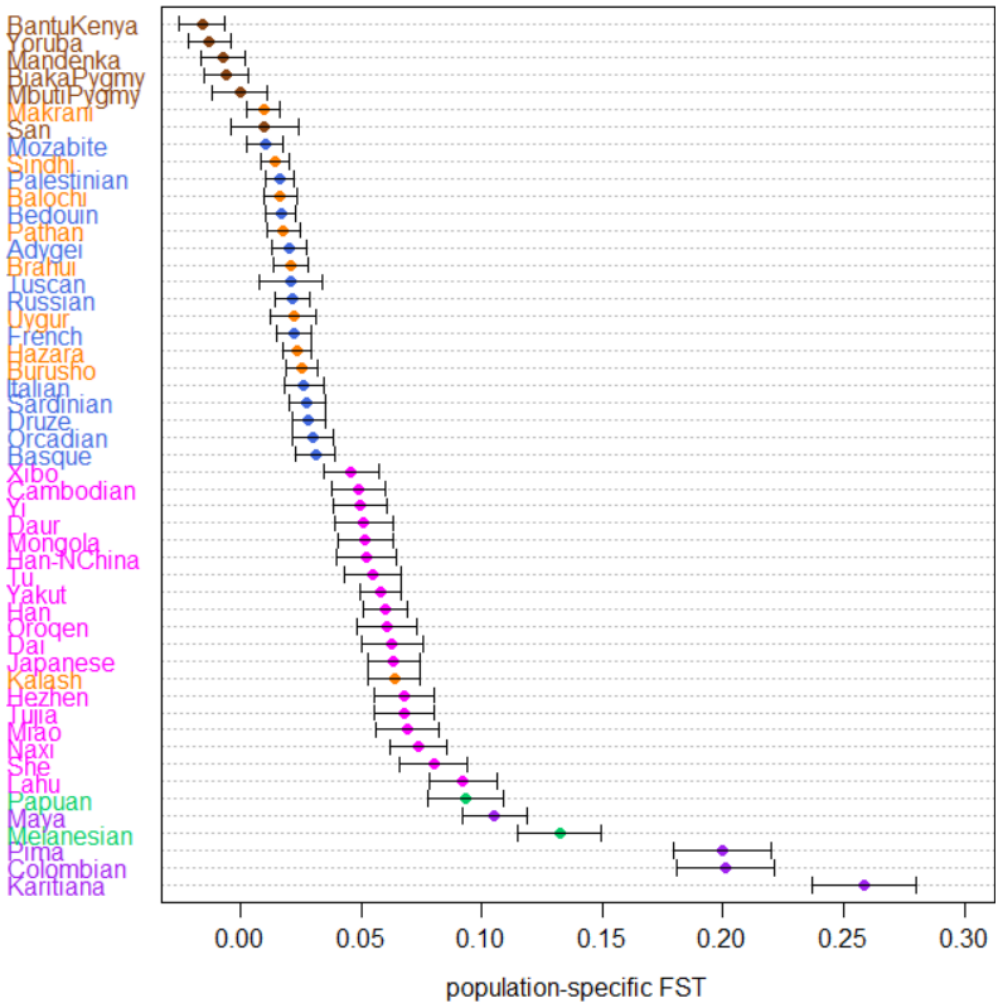


実行結果として、以下の図が得られます。**ここまでで、原著論文の Figure 2B の図の再現に成功したことになります。これは図 6.4b のカラー版に相当するものだから**です。ここまでで例題 6.1 の 2 の問いに全て答えたことになります。



198-207 行目は、例題 6.1 の 2 の問いとは直接関係ありませんが、前述のとおりいくつかの図をまず作成し、その中から最終的に論文の図としてどれを用いるのが適切かを判断します。

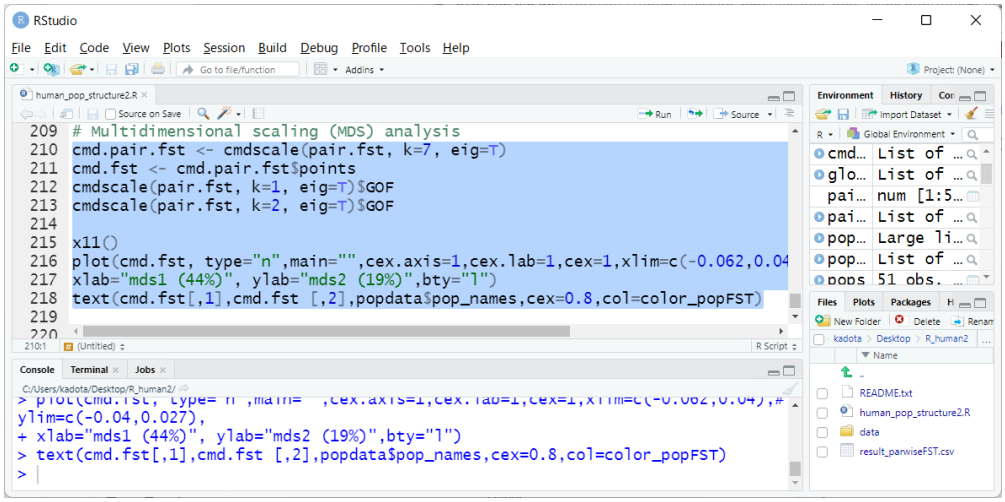
これは、集団固有 F_{ST} 値で色付けされた上図を、R スクリプトの 56-68 行目で定義した「サンプリング場所ごとにどの色を割り振るかを指定した情報」である `LocColors` オブジェクトを用いて（color オプションで与えて）dotchart 関数を実行するものです。実行結果として、以下の図が得られます。



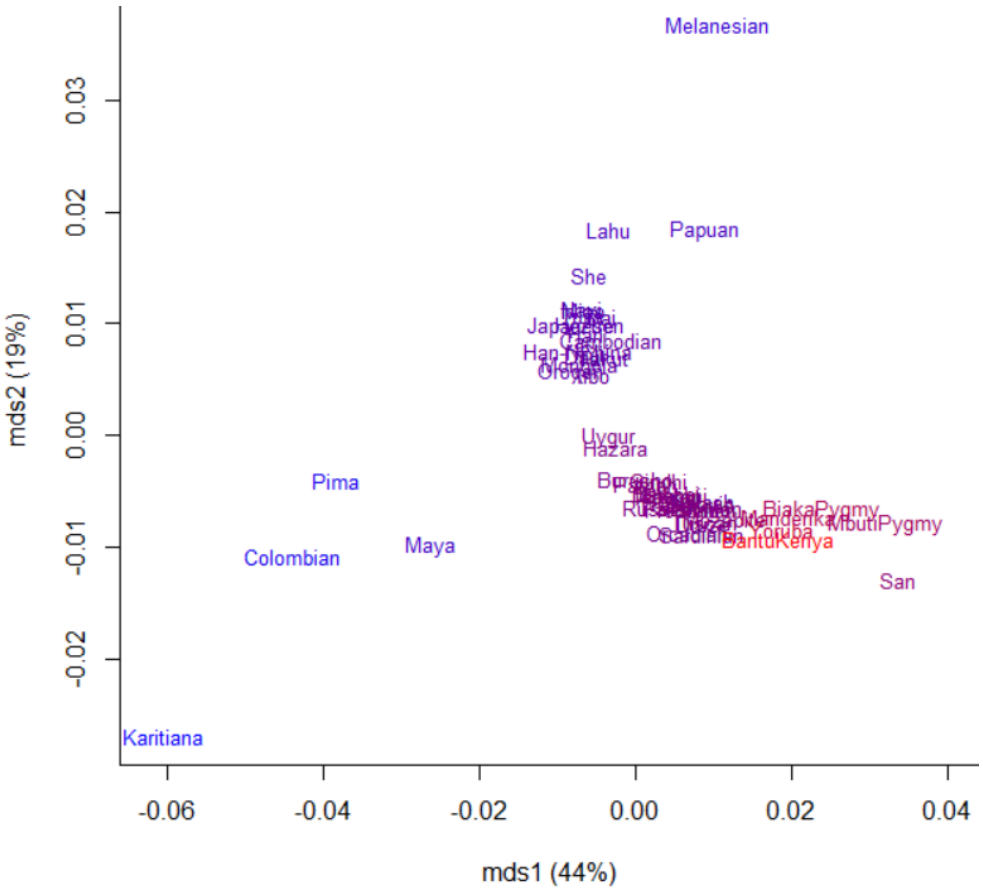
左側のサンプリング地点名のカラーリングと、集団固有 F_{ST} 値の分布が概ね一致していることがわかります。逆にいえば、これを確かめたいがために、このような図の描画も行います。

3 の解答例

MDS 解析の図は、原著論文の Figure 2D に相当します。R スクリプトの 210-218 行目を実行すれば得られます。問題文中で言及されている `pair.fst` オブジェクトは、R スクリプトの 34 行目で得ています。



実行結果として、以下の図が得られます。



これまで述べてきた通り、集団名の色は集団固有 F_{ST} の推定値の大きさに比例しています（赤が最も小さく、青は最も大きい）。プロットの右下付近に位置する、東アフリカの Bantu Kenya の集団固有 F_{ST} が最も小さく、祖先に近いことを示唆しています。一方、Karitiana をはじめとする中南米の集団固有 F_{ST} は最も大きく、進化の歴史が浅いことを示唆しています。原著論文では、ヒトの他、タイセイヨウタラと野生ポプラの SNPs に基づく集団構造が解析されています。生態学を学ぶ読者にも参考になるはずです。