practicaGuiada

Cargamos la biblioteca para series temporales.

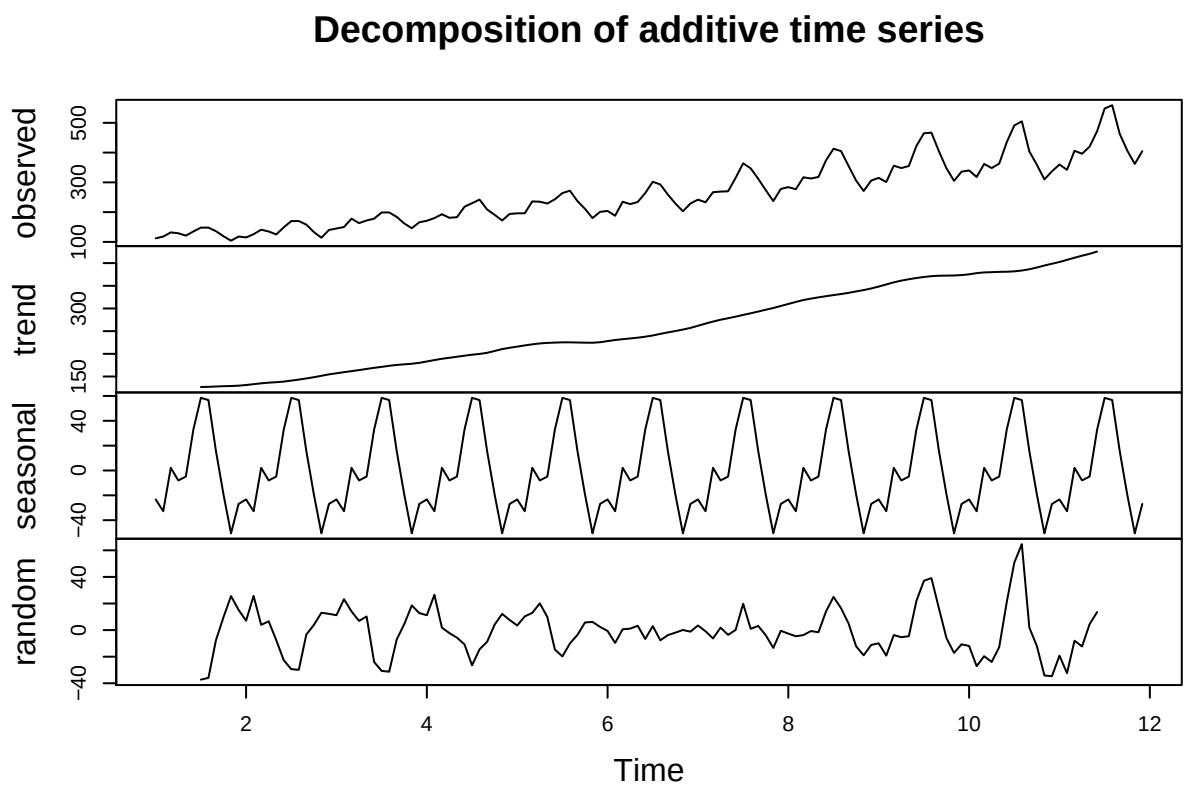
```
library(tseries)
```

Leemos los datos de la serie

```
serie = scan("pasajeros_1949_1959.dat")
```

Dividimos los datos en entrenamiento y test. Dejamos para ello el último año para la comprobación de la validez del modelo, y el resto lo usamos como train.

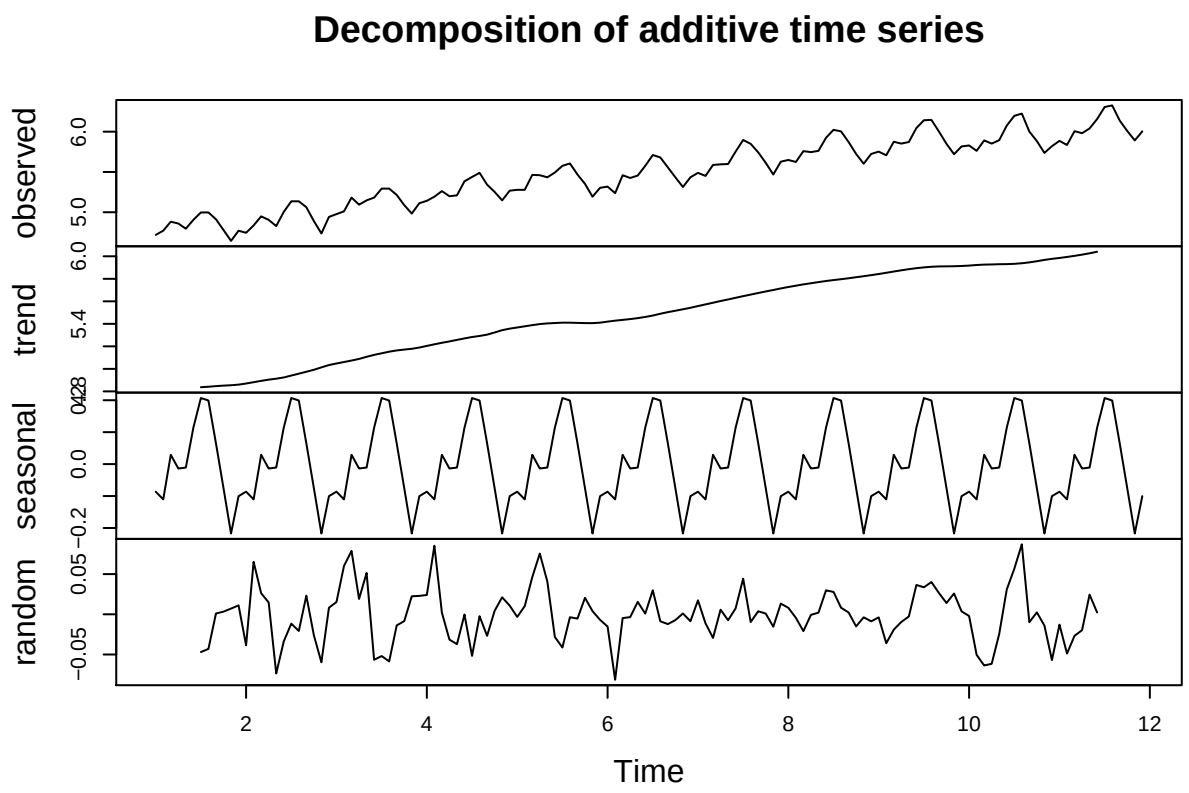
```
NTest = 12 # cantidad de datos a usar como test
NPred = 12 # cantidad de predicciones que queremos realizar
serie.ts = ts(serie, frequency = 12) # creamos el objeto serie temporal suponiendo una estacionalidad d
plot(decompose(serie.ts))
```



Observamos aquí: \ -observed: los valores de la serie\ -trend: la tendencia calculada mediante filtros\ -seasonal: estacionalidad en la que cada 12 instantes de tiempo se repite la serie\ -random: lo que queda de la serie una vez eliminadas tendencia y estacionalidad\

Como observamos en random, la varianza es alta, lo que puede dar problemas en un futuro para la estacionariedad, ya que como sabemos, una serie con estacionariedad no varía en media ni en varianza. Para ello, a la serie inicial le calculamos el logaritmo de la serie (calculamos el logaritmo tanto a los datos como a la serie temporal):

```r
serie.ts.log = log(serie.ts)
serie.log = log(serie)
plot(decompose(serie.ts.log))
```



Como observamos ahora, la varianza consta de menor variación a lo largo del tiempo, lo que en un futuro provocará que no tengamos problemas a la hora de calcular la estacionariedad.

Aplicando decompose sobre los datos podemos observar como los valores de cada mes en el atributo seasonal son exactamente los mismos, lo que nos hará falta para calcular la componente estacional y restarselo a la serie.

```r
decompose(serie.ts.log)
```

```
## $x
##          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1  4.718499 4.770685 4.882802 4.859812 4.795791 4.905275 4.997212 4.997212
## 2  4.744932 4.836282 4.948760 4.905275 4.828314 5.003946 5.135798 5.135798
## 3  4.976734 5.010635 5.181784 5.093750 5.147494 5.181784 5.293305 5.293305
## 4  5.141664 5.192957 5.262690 5.198497 5.209486 5.384495 5.438079 5.488938
## 5  5.278115 5.278115 5.463832 5.459586 5.433722 5.493061 5.575949 5.605802
## 6  5.318120 5.236442 5.459586 5.424950 5.455321 5.575949 5.710427 5.680173
## 7  5.488938 5.451038 5.587249 5.594711 5.598422 5.752573 5.897154 5.849325
## 8  5.648974 5.624018 5.758902 5.746203 5.762051 5.924256 6.023448 6.003887
## 9  5.752573 5.707110 5.874931 5.852202 5.872118 6.045005 6.142037 6.146329
## 10 5.828946 5.762051 5.891644 5.852202 5.894403 6.075346 6.196444 6.224558
## 11 5.886104 5.834811 6.006353 5.981414 6.040255 6.156979 6.306275 6.326149
##          Sep      Oct      Nov      Dec
```

```
## 1  4.912655 4.779123 4.644391 4.770685
## 2  5.062595 4.890349 4.736198 4.941642
## 3  5.214936 5.087596 4.983607 5.111988
## 4  5.342334 5.252273 5.147494 5.267858
## 5  5.468060 5.351858 5.192957 5.303305
## 6  5.556828 5.433722 5.313206 5.433722
## 7  5.743003 5.613128 5.468060 5.627621
## 8  5.872118 5.723585 5.602119 5.723585
## 9  6.001415 5.849325 5.720312 5.817111
## 10 6.001415 5.883322 5.736572 5.820083
## 11 6.137727 6.008813 5.891644 6.003887
##
## $seasonal
##            Jan         Feb         Mar         Apr         May         Jun
## 1  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 2  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 3  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 4  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 5  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 6  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 7  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 8  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 9  -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 10 -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
## 11 -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726  0.11403832
##            Jul         Aug         Sep         Oct         Nov         Dec
## 1   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 2   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 3   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 4   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 5   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 6   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 7   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 8   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 9   0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 10  0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
## 11  0.20689984  0.19914832  0.06503613 -0.07542627 -0.21722154 -0.10035385
##
## $trend
##         Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1        NA       NA       NA       NA       NA       NA 4.837280 4.841114
## 2  4.869840 4.881389 4.893411 4.904293 4.912752 4.923701 4.940483 4.957406
## 3  5.047776 5.060902 5.073812 5.088378 5.106906 5.124312 5.138282 5.152751
## 4  5.203909 5.218093 5.231553 5.243722 5.257413 5.270736 5.282916 5.292150
## 5  5.367695 5.378309 5.388417 5.397805 5.403849 5.407220 5.410364 5.410294
## 6  5.419628 5.428330 5.435128 5.442237 5.450659 5.461103 5.473655 5.489713
## 7  5.557864 5.572693 5.587498 5.602730 5.616658 5.631189 5.645937 5.659812
## 8  5.727153 5.738856 5.750676 5.760658 5.770846 5.780430 5.788745 5.796524
## 9  5.842665 5.853541 5.864863 5.875490 5.885654 5.894475 5.901555 5.907026
## 10 5.917360 5.922887 5.926146 5.927563 5.929657 5.930458 5.932964 5.938377
## 11 5.985269 5.994078 6.003991 6.014899 6.026589 6.040709       NA       NA
##         Sep      Oct      Nov      Dec
## 1  4.846596 4.851238 4.854488 4.859954
## 2  4.974380 4.991942 5.013095 5.033804
```

```
## 3  5.163718 5.171454 5.178401 5.189431
## 4  5.304079 5.323338 5.343560 5.357427
## 5  5.408381 5.406761 5.406218 5.410571
## 6  5.503974 5.516367 5.529403 5.542725
## 7  5.674172 5.687636 5.700766 5.714738
## 8  5.804821 5.814072 5.823075 5.832692
## 9  5.910012 5.910708 5.911637 5.913829
## 10 5.946188 5.956352 5.967813 5.977291
## 11       NA       NA       NA       NA
##
## $random
##              Jan           Feb           Mar           Apr           May
## 1             NA            NA            NA            NA            NA
## 2  -0.0386339970  0.0653225337  0.0261932467  0.0147482216 -0.0736314277
## 3   0.0152313141  0.0601629324  0.0788155515  0.0191380326  0.0513961090
## 4   0.0240281013  0.0852933536  0.0019817093 -0.0314592700 -0.0371192467
## 5  -0.0033066372  0.0102350395  0.0462589183  0.0755466804  0.0406806494
## 6  -0.0152342638 -0.0814587149 -0.0046979403 -0.0035211005  0.0154697303
## 7   0.0173472978 -0.0112246683 -0.0294051083  0.0057470561 -0.0074287676
## 8   0.0080946404 -0.0044088265 -0.0209297699 -0.0006885087  0.0020126383
## 9  -0.0038180964 -0.0360012050 -0.0190885497 -0.0095213382 -0.0027287804
## 10 -0.0021409158 -0.0504060472 -0.0636581078 -0.0615944124 -0.0244470388
## 11 -0.0128909429 -0.0488378968 -0.0267934492 -0.0197188603  0.0244726346
##              Jun           Jul           Aug           Sep           Oct
## 1             NA -0.0469674232 -0.0430504999  0.0010228263  0.0033113203
## 2  -0.0337929898 -0.0115840295 -0.0207556255  0.0231791098 -0.0261668558
## 3  -0.0565667896 -0.0518768430 -0.0585941210 -0.0138188466 -0.0084314293
## 4  -0.0002791708 -0.0517365147 -0.0023602730 -0.0267807069  0.0043612826
## 5  -0.0281966990 -0.0413144046 -0.0036404389 -0.0053568900  0.0205235777
## 6   0.0008077175  0.0298726519 -0.0086891533 -0.0121823358 -0.0072186807
## 7   0.0073449512  0.0443173338 -0.0096358733  0.0037950438  0.0009179914
## 8   0.0297872004  0.0278023678  0.0082145728  0.0022607399 -0.0150607236
## 9   0.0364915439  0.0335829803  0.0401549408  0.0263671358  0.0140430437
## 10  0.0308492379  0.0565803816  0.0870329719 -0.0098095757  0.0023969741
## 11  0.0022314988            NA            NA            NA            NA
##              Nov           Dec
## 1   0.0071245946  0.0110841849
## 2  -0.0596746089  0.0081925943
## 3   0.0224266863  0.0229108650
## 4   0.0211557510  0.0107849852
## 5   0.0039607447 -0.0069125020
## 6   0.0010246364 -0.0086488767
## 7  -0.0154847585  0.0132371681
## 8  -0.0037344751 -0.0087532126
## 9   0.0258967636  0.0036357185
## 10 -0.0140188337 -0.0568544240
## 11            NA            NA
##
## $figure
##  [1] -0.08627390 -0.11042950  0.02915584 -0.01376615 -0.01080726
##  [6]  0.11403832  0.20689984  0.19914832  0.06503613 -0.07542627
## [11] -0.21722154 -0.10035385
##
## $type
```

```
## [1] "additive"
##
## attr(,"class")
## [1] "decomposed.ts"
```

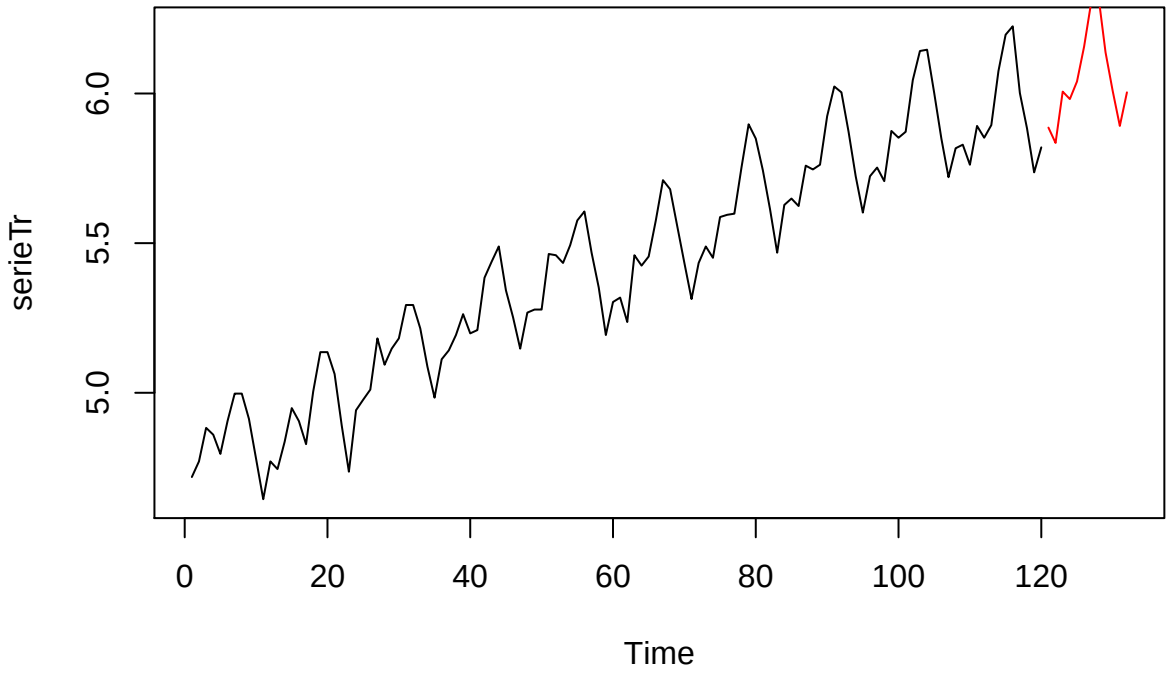
Procedemos a la división de los datos en train y test

```
serieTr = serie.log[1:(length(serie.log)-NTest)] # tomamos todos los valores de la serie
                                                 # como train excepto los 12 últimos
tiempoTr = 1:length(serieTr) # instantes de tiempo de esta serie train

# mismo proceso para la serie de test...
serieTs = serie.log[(length(serie.log)-NTest+1):length(serie)]
tiempoTs = (tiempoTr[length(tiempoTr)]+1):(tiempoTr[length(tiempoTr)]+NTest)
```

Representamos la serie de entrenamiento y la linea de la serie de test en rojo con los parametros necesarios para que salga de forma correcta dentro de los límites de la gráfica y en el lugar adecuado.

```
plot.ts(serieTr, xlim=c(1, tiempoTs[length(tiempoTs)]))
lines(tiempoTs, serieTs, col="red")
```



A continuación prodecemos a modelar la tendencia. Parece ser que la tendencia es lineal y creciente, por lo que tendrá la forma de:\ serie = parametroA*tiempo + parametroB.\ Con la funcion lm calculamos esos dos parametros para modelar dicha tendencia.

```
parametros.H1 = lm(serieTr ~tiempoTr)
parametros.H1
```

```
##
```

```
## Call:
## lm(formula = serieTr ~ tiempoTr)
##
## Coefficients:
## (Intercept)     tiempoTr
##     4.79025      0.01058
```

Intercept es el termino independiente (parametroB) y el otro es el que multiplica al tiempo para poder calcular la serie (parametroA). Para modelar la tendencia usamos la fórmula descrita antes:

```
tendEstimadaTr = parametros.H1$coefficients[1] + tiempoTr*parametros.H1$coefficients[2] # tendencia en
tendEstimadaTs = parametros.H1$coefficients[1] + tiempoTs*parametros.H1$coefficients[2] # tendencia en
```

Comprobamos de forma visual si la tendencia se ajusta al modelo que tenemos de la serie temporal.

```
plot.ts(serieTr, xlim=c(1, tiempoTs[length(tiempoTs)]))
lines(tiempoTs, serieTs, col="red")
lines(tiempoTr, tendEstimadaTr, col = "blue")
lines(tiempoTs, tendEstimadaTs, col = "green")
```