YVR monthly average temperature: forecast rules and h-step ahead intervals

2026 January 20

```
source("simple-fcrules.R")
source("esm-fcrules.R")
#source("linear-fcrules.R")
# The formats of forecast functions are the following.
# esm_fc = function(train,holdout,alpha,level,iprint) # simple exponential smoothing
# lholt_fc = function(train,holdout,alpha,beta,level,slope,iprint) # Holt linear
# Winters multiplicative seasonal and additive seasonal
# mseason_fc = function(train,holdout,alpha,beta,gamma,level,slope,season,iprint)
# aseason_fc = function(train,holdout,alpha,beta,gamma,level,slope,season,iprint)

# Some of these functions are to be coded in the labs and submitted on canvas.
```

Compare forecast rules for Vancouver monthly total precipitation

```
v = read.csv("vanc-prec-temp.csv",header=T)
print(names(v))
#> [1] "yearmon"   "totprecip" "meantemp"
# length is 86*12 = 1032
print(nrow(v))
#> [1] 1032
nn = nrow(v)
summary(v$meantemp)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  -6.330   5.390   9.655  10.137  14.990  20.600

# summaries for monthly average temperature by month
month = v$yearmon %% 100

print(tapply(v$meantemp, month, summary))
#> $`1`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  -6.330   2.322   3.650   3.327   4.640   7.220
#>
#> $`2`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   0.400   3.748   4.615   4.560   5.497   7.570
#>
#> $`3`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   3.290   5.640   6.290   6.372   7.242   8.500
#>
#> $`4`
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   7.010   8.617   9.155   9.142   9.725  11.780
#>
#> $`5`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   10.55   11.80   12.54   12.64   13.40   15.18
#>
#> $`6`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   13.54   14.71   15.35   15.45   16.14   18.02
#>
#> $`7`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   15.62   17.12   17.66   17.73   18.43   20.60
#>
#> $`8`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   15.72   16.94   17.65   17.64   18.44   19.78
#>
#> $`9`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   12.39   13.93   14.64   14.65   15.43   16.60
#>
#> $`10`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   7.850   9.627  10.155  10.204  10.810  13.050
#>
#> $`11`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   0.080   5.143   6.140   6.149   7.185   9.260
#>
#> $`12`
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>   0.600   2.630   4.210   3.775   4.957   7.010

print(tapply(v$meantemp, month, sd))
#>         1         2         3         4         5         6         7         8
#> 2.2143486 1.5010212 1.1610470 0.9448122 1.0835425 1.0432779 1.0154016 0.9794246
#>         9        10        11        12
#> 0.9738423 0.9114715 1.4713069 1.6303379


# holdout set: last 18 years
ntrain = 12*68
# change to degree Kelvin so that multiplicative rule shoukd
vtrain = v$meantemp[1:ntrain] + 273
vholdout = v$meantemp[(ntrain+1):nn] + 273
mon_holdout = v$yearmon[(ntrain+1):nn]

z = ts(vtrain,start=c(1938,1),frequency=12)

# Winters additive seasonal
wafit = HoltWinters(z,seasonal="additive")
```

```r
# trend column means estimated slope
print(wafit$fitted[(ntrain-24):(ntrain-12),])
#>           xhat    level       trend      season
#>  [1,] 277.7036 284.1201 0.001056235 -6.41755509
#>  [2,] 277.9547 284.1887 0.001056235 -6.23509711
#>  [3,] 278.5496 284.0597 0.001056235 -5.51118849
#>  [4,] 280.2892 283.9302 0.001056235 -3.64203486
#>  [5,] 283.1664 284.0431 0.001056235 -0.87775169
#>  [6,] 286.3415 284.0393 0.001056235  2.30106472
#>  [7,] 289.5568 284.1447 0.001056235  5.41099797
#>  [8,] 291.6546 284.0458 0.001056235  7.60776445
#>  [9,] 291.6109 283.9910 0.001056235  7.61880998
#> [10,] 288.5470 284.0296 0.001056235  4.51636952
#> [11,] 283.8535 283.9390 0.001056235 -0.08660681
#> [12,] 280.0000 283.9857 0.001056235 -3.98676154
#> [13,] 277.4958 283.8457 0.001056235 -6.35097409


print(wafit)
#> Holt-Winters exponential smoothing with trend and additive seasonal component.
#>
#> Call:
#> HoltWinters(x = z, seasonal = "additive")
#>
#> Smoothing parameters:
#>  alpha: 0.1044861
#>  beta : 0
#>  gamma: 0.1150224
#>
#> Coefficients:
#>              [,1]
#> a   283.857629893
#> b     0.001056235
#> s1   -6.363302431
#> s2   -5.639900083
#> s3   -3.531740405
#> s4   -0.882535606
#> s5    2.403918094
#> s6    5.312445001
#> s7    7.552694490
#> s8    7.655801063
#> s9    4.426033148
#> s10  -0.041641446
#> s11  -4.125814513
#> s12  -6.340237418

# $fitted is missing for first year
names(wafit)
#> [1] "fitted"       "x"            "alpha"        "beta"         "gamma"
#> [6] "coefficients" "seasonal"     "SSE"          "call"

print(sqrt(wafit$SSE/(ntrain-12)))
#> [1] 1.301437
```

```r
# Winters multiplicative seasonal
wmfit = HoltWinters(z,seasonal="multiplicative")
# trend column means estimated slope
print(wmfit$fitted[(ntrain-24):(ntrain-12),])
#>           xhat    level       trend    season
#>  [1,] 277.6781 284.1139 0.001056235 0.9773442
#>  [2,] 277.9302 284.1850 0.001056235 0.9779867
#>  [3,] 278.5327 284.0589 0.001056235 0.9805419
#>  [4,] 280.2798 283.9318 0.001056235 0.9871338
#>  [5,] 283.1609 284.0444 0.001056235 0.9968857
#>  [6,] 286.3434 284.0413 0.001056235 1.0081011
#>  [7,] 289.5639 284.1431 0.001056235 1.0190737
#>  [8,] 291.6690 284.0478 0.001056235 1.0268268
#>  [9,] 291.6267 283.9943 0.001056235 1.0268711
#> [10,] 288.5553 284.0295 0.001056235 1.0159305
#> [11,] 283.8553 283.9417 0.001056235 0.9996919
#> [12,] 279.9927 283.9871 0.001056235 0.9859309
#> [13,] 277.4887 283.8493 0.001056235 0.9775879

print(wmfit)
#> Holt-Winters exponential smoothing with trend and multiplicative seasonal component.
#>
#> Call:
#> HoltWinters(x = z, seasonal = "multiplicative")
#>
#> Smoothing parameters:
#>  alpha: 0.1019261
#>  beta : 0
#>  gamma: 0.1147524
#>
#> Coefficients:
#>             [,1]
#> a   2.838620e+02
#> b   1.056235e-03
#> s1  9.775440e-01
#> s2  9.800945e-01
#> s3  9.875258e-01
#> s4  9.968708e-01
#> s5  1.008463e+00
#> s6  1.018724e+00
#> s7  1.026628e+00
#> s8  1.026996e+00
#> s9  1.015609e+00
#> s10 9.998497e-01
#> s11 9.854434e-01
#> s12 9.776283e-01

print(sqrt(wmfit$SSE/(ntrain-12)))
#> [1] 1.302265

alpha = wafit$alpha;  beta = wafit$beta; gamma = wafit$gamma
level = wafit$coef[1];  slope = wafit$coef[2]; season = wafit$coefficient[3:14]
aseason = aseason_fc(vtrain,vholdout,alpha,beta,gamma,level,slope,season,iprint=F)
```

```
alph = wmfit$alpha;  bet = wmfit$beta; gamm = wmfit$gamma
leve = wmfit$coef[1];  slop = wmfit$coef[2]; seaso = wmfit$coefficient[3:14]
mseason = mseason_fc(vtrain,vholdout,alph,bet,gamm,leve,slop,seaso,iprint=F)

persbm = persistbymonth_fc(vtrain,vholdout,iprint=F)
iidbm = iidbymonth_fc(vtrain,vholdout,iprint=F)

out = cbind(mon_holdout/100,vholdout, aseason$fc, mseason$fc, persbm$fc, iidbm$fc)
colnames(out) = c("yearmon","holdout","add_seasonal","mult_seasonal","persist_mon","iid_bymon")
print(round(out[1:12,],2))
#>       yearmon holdout add_seasonal mult_seasonal persist_mon iid_bymon
#>  [1,] 2006.01  279.30       277.50        277.49      276.71    276.06
#>  [2,] 2006.02  277.28       278.41        278.40      277.30    277.58
#>  [3,] 2006.03  279.55       280.40        280.40      281.36    279.30
#>  [4,] 2006.04  282.30       282.96        282.96      283.12    282.13
#>  [5,] 2006.05  286.04       286.18        286.19      287.34    285.49
#>  [6,] 2006.06  289.67       289.07        289.08      288.60    288.31
#>  [7,] 2006.07  291.68       291.38        291.39      291.12    290.52
#>  [8,] 2006.08  290.56       291.51        291.52      291.97    290.42
#>  [9,] 2006.09  288.26       288.19        288.20      287.67    287.48
#> [10,] 2006.10  283.04       283.73        283.73      284.29    283.14
#> [11,] 2006.11  278.74       279.57        279.57      278.65    279.09
#> [12,] 2006.12  277.47       277.27        277.27      277.60    276.83

rmse_vec = c(aseason$rmse, mseason$rmse, persbm$rmse, iidbm$rmse)

cat(round(rmse_vec,2), "\n")
#> 1.22 1.22 1.63 1.34

# Interpret: how do the methods compare?
```

1-step to h-step ahead forecast intervals at end of training set

```
class(wafit)
#> [1] "HoltWinters"

class(wmfit)
#> [1] "HoltWinters"

# predict method: see help(predict.HoltWinters)

wa_pred = predict(wafit, n.ahead=14, prediction.interval=T, level=0.90)

wm_pred = predict(wmfit, n.ahead=14, prediction.interval=T, level=0.90)


mon_ahead = v$yearmon[(ntrain+1):(ntrain+14)]

pred_df = as.data.frame(cbind(mon_ahead/100,wa_pred,wm_pred))

names(pred_df) = c("yearmon","pt_add","upr_add","lwr_add","pt_mul","upr_mul","lwr_mul")
```

```
print(round(pred_df,3))
#>    yearmon  pt_add upr_add lwr_add  pt_mul upr_mul  lwr_mul
#> 1  2006.01 277.495 279.637 275.353 277.489 277.928  277.049
#> 2  2006.02 278.220 280.373 276.066 278.214 278.705  277.723
#> 3  2006.03 280.329 282.494 278.164 280.324 280.863  279.785
#> 4  2006.04 282.979 285.156 280.803 282.978 283.562  282.394
#> 5  2006.05 286.267 288.455 284.079 286.270 286.896  285.643
#> 6  2006.06 289.176 291.376 286.977 289.183 289.851  288.516
#> 7  2006.07 291.418 293.629 289.207 291.428 292.134  290.722
#> 8  2006.08 291.522 293.744 289.300 291.534 292.273  290.795
#> 9  2006.09 288.293 290.527 286.060 288.303 289.067  287.538
#> 10 2006.10 283.827 286.071 281.582 283.830 284.617  283.043
#> 11 2006.11 279.743 281.999 277.488 279.741 280.550  278.933
#> 12 2006.12 277.530 279.797 275.263 277.524 684.176 -129.129
#> 13 2007.01 277.508 279.818 275.198 277.501 479.691   75.311
#> 14 2007.02 278.233 280.553 275.912 278.226 480.943   75.509

cv = qnorm(0.95)
SE_add = (pred_df$upr_add - pred_df$lwr_add)/(2*cv)
SE_add = round(SE_add,3)
cbind(pred_df$yearmon, SE_add)
#>               SE_add
#>  [1,] 2006.01  1.302
#>  [2,] 2006.02  1.309
#>  [3,] 2006.03  1.316
#>  [4,] 2006.04  1.323
#>  [5,] 2006.05  1.330
#>  [6,] 2006.06  1.337
#>  [7,] 2006.07  1.344
#>  [8,] 2006.08  1.351
#>  [9,] 2006.09  1.358
#> [10,] 2006.10  1.365
#> [11,] 2006.11  1.371
#> [12,] 2006.12  1.378
#> [13,] 2007.01  1.404
#> [14,] 2007.02  1.411

# Note: SEs of prediction intervals increase with longer forecast horizon
```

Moving 1-step ahead forecasts for holdout set

```
# Use the training and holdout sets together as a ts object
# z_all = ts(v$meantemp,start=c(1938,1),frequency=12)
# wafit_all = HoltWinters(z_all, alpha=wafit$alpha, ... )

# Exercise: fill in the other arguments and extract suitable output.
# Make a second application of HoltWinters with inputs (a) the entire data set
# (b) some output components of HoltWinters applied to the training set
# and (c) ??
# Extract part of the output of the second application to match results of
# R functions based on pseudocodes in the slides.
```