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

2026 January 14

```
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
vtrain = v$meantemp[1:ntrain]
vholdout = v$meantemp[(ntrain+1):nn]
mon_holdout = v$yearmon[(ntrain+1):nn]

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

# Winters additive seasonal
wafit = HoltWinters(z,seasonal="additive")
# trend column means estimated slope
```

```r
print(wafit$fitted[(ntrain-24):(ntrain-12),])
#>            xhat    level       trend      season
#>  [1,]  4.703609 11.12011 0.001056235 -6.41755509
#>  [2,]  4.954662 11.18870 0.001056235 -6.23509711
#>  [3,]  5.549577 11.05971 0.001056235 -5.51118849
#>  [4,]  7.289223 10.93020 0.001056235 -3.64203486
#>  [5,] 10.166444 11.04314 0.001056235 -0.87775169
#>  [6,] 13.341464 11.03934 0.001056235  2.30106472
#>  [7,] 16.556786 11.14473 0.001056235  5.41099797
#>  [8,] 18.654638 11.04582 0.001056235  7.60776445
#>  [9,] 18.610878 10.99101 0.001056235  7.61880998
#> [10,] 15.547017 11.02959 0.001056235  4.51636952
#> [11,] 10.853461 10.93901 0.001056235 -0.08660681
#> [12,]  6.999974 10.98568 0.001056235 -3.98676154
#> [13,]  4.495765 10.84568 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   10.857629893
#> 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,]  3.639903 17.22127 0.001056235 0.2113479
#>  [2,]  6.348757 24.97634 0.001056235 0.2541801
#>  [3,]  4.707004 15.02880 0.001056235 0.3131768
#>  [4,]  6.604964 13.78445 0.001056235 0.4791238
#>  [5,] 12.854264 17.29579 0.001056235 0.7431566
#>  [6,] 15.570335 13.77099 0.001056235 1.1305751
#>  [7,] 18.923452 12.72918 0.001056235 1.4864960
#>  [8,] 18.299389 10.58770 0.001056235 1.7281910
#>  [9,] 18.135242 10.48928 0.001056235 1.7287570
#> [10,] 15.854939 10.95307 0.001056235 1.4473943
#> [11,] 10.440670 10.16959 0.001056235 1.0265493
#> [12,]  4.318850 10.96351 0.001056235 0.3938915
#> [13,]  3.025876 14.20315 0.001056235 0.2130268

print(wmfit)
#> Holt-Winters exponential smoothing with trend and multiplicative seasonal component.
#>
#> Call:
#> HoltWinters(x = z, seasonal = "multiplicative")
#>
#> Smoothing parameters:
#>  alpha: 0.9583052
#>  beta : 0
#>  gamma: 0.5880997
#>
#> Coefficients:
#>             [,1]
#> a   21.285430666
#> b    0.001056235
#> s1   0.249874718
#> s2   0.312452839
#> s3   0.481612003
#> s4   0.738288000
#> s5   1.128205057
#> s6   1.478799053
#> s7   1.727771627
#> s8   1.730625754
#> s9   1.444537202
#> s10  1.028448935
#> s11  0.396189598
#> s12  0.214840184

print(sqrt(wmfit$SSE/(ntrain-12)))
#> [1] 2.134092

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    6.30         4.50          5.32        3.71      3.06
#>  [2,] 2006.02    4.28         5.41          7.83        4.30      4.58
#>  [3,] 2006.03    6.55         7.40          6.83        8.36      6.30
#>  [4,] 2006.04    9.30         9.96         10.06       10.12      9.13
#>  [5,] 2006.05   13.04        13.18         14.26       14.34     12.49
#>  [6,] 2006.06   16.67        16.07         17.16       15.60     15.31
#>  [7,] 2006.07   18.68        18.38         19.50       18.12     17.52
#>  [8,] 2006.08   17.56        18.51         18.75       18.97     17.42
#>  [9,] 2006.09   15.26        15.19         14.70       14.67     14.48
#> [10,] 2006.10   10.04        10.73         10.85       11.29     10.14
#> [11,] 2006.11    5.74         6.57          3.88        5.65      6.09
#> [12,] 2006.12    4.47         4.27          3.07        4.60      3.83

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

cat(round(rmse_vec,2), "\n")
#> 1.22 1.84 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  4.495   6.637   2.353  5.319   8.745    1.893
#> 2  2006.02  5.220   7.373   3.066  6.651  12.054    1.248
#> 3  2006.03  7.329   9.494   5.164 10.253  19.182    1.324
#> 4  2006.04  9.979  12.156   7.803 15.718  29.779    1.657
#> 5  2006.05 13.267  15.455  11.079 24.020  45.748    2.292
#> 6  2006.06 16.176  18.376  13.977 31.486  60.154    2.818
#> 7  2006.07 18.418  20.629  16.207 36.789  70.447    3.131
#> 8  2006.08 18.522  20.744  16.300 36.852  70.730    2.973
#> 9  2006.09 15.293  17.527  13.060 30.761  59.239    2.284
#> 10 2006.10 10.827  13.071   8.582 21.902  42.457    1.347
#> 11 2006.11  6.743   8.999   4.488  8.438  17.061   -0.185
#> 12 2006.12  4.530   6.797   2.263  4.576 701.514 -692.363
#> 13 2007.01  4.508   6.818   2.198  5.322 795.697 -785.053
#> 14 2007.02  5.233   7.553   2.912  6.655 994.975 -981.665

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.
```