KALMAN FILTER AND MEASURE THEORY EXERCISE

(1) We model three producer price index time series, indexes for "Crude Materials for Further Processing", "Intermediate Materials and Parts", and "Finished Goods". $p_t = [p_{1t}, p_{2t}, p_{3t}]$ is the vector of base-e logarithms of these three time series. The data on them, monthly starting in April 1947, are available on the course web site as a bare text file (`ppi3.txt`) suitable for reading in to Matlab and as part of the `kfex.RData` file, which can be read in to R with the `load()` command. We define $\bar{p}_t$ as an unobservable aggregate price index and $\tilde{p}_t = p_t - \bar{p}_t$. That is, for each i $\tilde{p}_{it} = p_{it} - \bar{p}_t$. We specify the model as

$$\begin{aligned}\tilde{p}_{it} &= \alpha_{i1}\tilde{p}_{i,t-1} + \alpha_{i,2}\tilde{p}_{i,t-2} + \varepsilon_{it}\\ \bar{p}_t &= \gamma + \beta_1\bar{p}_{t-1} + \beta_2\bar{p}_{t-2} + \nu_t\,.\end{aligned}$$

The ε_{it}'s and ν_t's are Gaussian, mean 0, and independent of each other at all date pairs. The variance of ε_{it} is σ_i^2 for all t and the variance of ν_t is τ^2 for all t. Fitted values for all these parameters are in the `kfex.RData` file with self-explanatory names. They are also available as separate unlabeled text files suitable for reading into Matlab. The `alpha.txt` file is a 4×2 matrix with the three α pairs in the first three rows and β in the bottom row. The `omega.txt` file contains the three σ_i's followed by τ, followed by γ. (Note — standard errors here, not variances.)

To start the Kalman filter you need an initial distribution for the state. Use zero as the initial mean of $\tilde{p}$ and $\bar{p}$, and as a variance matrix for $\tilde{p}$ and $\bar{p}$ use a diagonal matrix with the four elements in `sig0.txt` (for Matlab) or the variable `sig0` in `kfex.RData` as the diagonal elements (with $\bar{p}$ at the bottom). These numbers (other than `sig0`) come from maximizing likelihood for the model. Note that the state will not be just $\tilde{p}, \bar{p}$, so the variance matrix for these four variables is only an input into your construction of the full covariance matrix of the initial state.

(a) Put this model into standard state-space form, distinguishing plant and observation equations.

Here's one way to do it: Let s_t be the 9-element vector consisting of the 3-element vector $\tilde{p}_t$, $\bar{p}_t$, the lagged values of these four, and κ_t, which is our name for the component of the state that is always 1. Then the state equation is

$$s_t = \begin{bmatrix} A_1 & A_2 & G \\ I & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} s_{t-1} + \begin{bmatrix} \varepsilon_t \\ \nu_t \\ 0 \\ 4\times 1 \\ 0 \end{bmatrix},$$

Date: October 11, 2008.

where A_s is a diagonal matrix with $\alpha_{1s}, \alpha_{2s}, \alpha_{3s}, \beta_s$ on the diagonal and G is a 4×1 vector of three zeros over γ. The observation equation is

$$y_t = \begin{bmatrix} I & 1 & 0 \\ 3\times 3 & 3\times 1 & 3\times 5 \end{bmatrix} s_t \, .$$

(b) Using the Kalman filter, estimate the time path of the unobservable $\bar{p}$. Plot the three data time series and the $\bar{p}_t$ time series on the same graph (using different colors or otherwise distinguishing the four series). Also plot the filtered time paths for the three elements of $\tilde{p}$. (On a separate graph, since these will have mean zero and the others will not.) Is the accuracy of the estimate of $\bar{p}$ stable through time? Indicate on the graph a one-standard error band around the estimated $\bar{p}$ at a few points in time. (Not at every point, as this will clutter up the graph too much.)

(c) Find Kalman smoothed estimates of $\bar{p}$ and plot on one graph the smoothed and filtered estimates of $\bar{p}$.

To do this exercise you will need a Kalman filtering and a Kalman smoothing program. These are not very hard to write from scratch, but you will save yourself time by using prewritten programs. A bare-bones program that does one iteration of the Kalman filter, and a corresponding one for the Kalman smoother, are available in Matlab and R versions on the course web site. To use these you have to write your own initialization of the filter and loop to iterate it. You can use other programs if you like.

The requested plots are at the back of this document, along with R code that produced the filtered and smoothed estimates and the graphs. The error bands and $\bar{p}$ itself are plotted on a separate graph. Interesting points: The filtered estimates of $\bar{p}$ almost coincide with the finished goods index after the early part of the sample. The smoothed estimates, after deviating from the filtered ones at the beginning of the sample, almost coincide with them until the '90's, when the data series become more volatile. If you printed out the covariance matrix of the state, you would have seen that it is clearly rank 1 — we have four shocks and a three-dimensional observation vector, so there is only one dimension of variability left. The code for the Kalman filter has all the parameters packed into `param`. This carries over from my having maximized the likelihood, where `csminwel` needs all parameters in one vector. Also, because the variances have to stay positive, the `param` vector contains their logs. I might have had you maximize he likelihood yourselves, but the likelihood was very ill conditioned and the optimization converged only with considerable tweaking. This explains why `param` contains the sums and differences of each variable's autoregressive parameters, rather than those parameters themselves — this made the parameters more independent and thereby aided convergence.

(2) The stochastic process X_t has a state space S consisting of the functions $\{1,2,3,4,5\} \mapsto \mathbb{R}$

$$\begin{aligned} \omega_1 &= \{1,4,6,8,10\} \\ \omega_2 &= \{1,4,6,8,-14\} \\ \omega_3 &= \{1,0,0,0,0\} \\ \omega_4 &= \{1,0,0,2,40\} \end{aligned}$$

$$\omega_5 = \{1, 4, 6, 4, 2\} \ .$$

The probability measure P on this space gives equal probability .20 to each of the 5 time paths.

(a) Find the marginal probability distributions for the random variables X_t at each of the 5 dates. I should have been explicit that we are in the common special case where the points in S are just the time paths of X_t, but people seemed to figure that out. X_1 is one with probability 1. X_2 is 4 w.p. .6, 0 w.p. .4. X_3 is 6 w.p. .6, 0 w.p. .4, X_4 is 0, 2, or 4, each w.p. .2, or 8 w.p. .4. X_5 is 10, -14, 0, 40, or 2, each w.p. .2.

(b) Define $\mathcal{F}_t$ as the σ-field generated by the random variables $\{X_s, s < t\}$. Enumerate the members of the σ-field $\mathcal{F}_t$ for each of the dates $t = 1, 2, 3$.

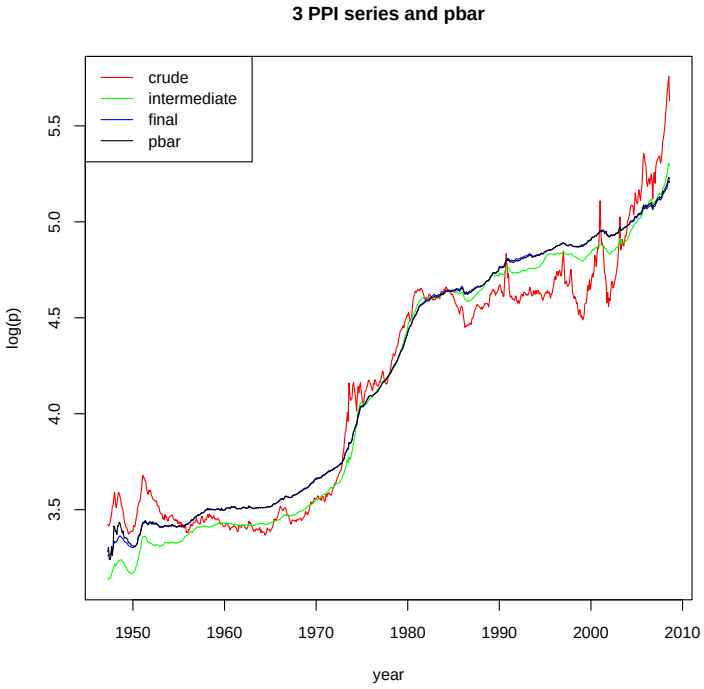
$\mathcal{F}_1$:	$S, \emptyset$
$\mathcal{F}_2$:	$S, \emptyset, \{\omega_1, \omega_2, \omega_5\}, \{\omega_3, \omega_4\}$
$\mathcal{F}_3$:	same as $\mathcal{F}_2$

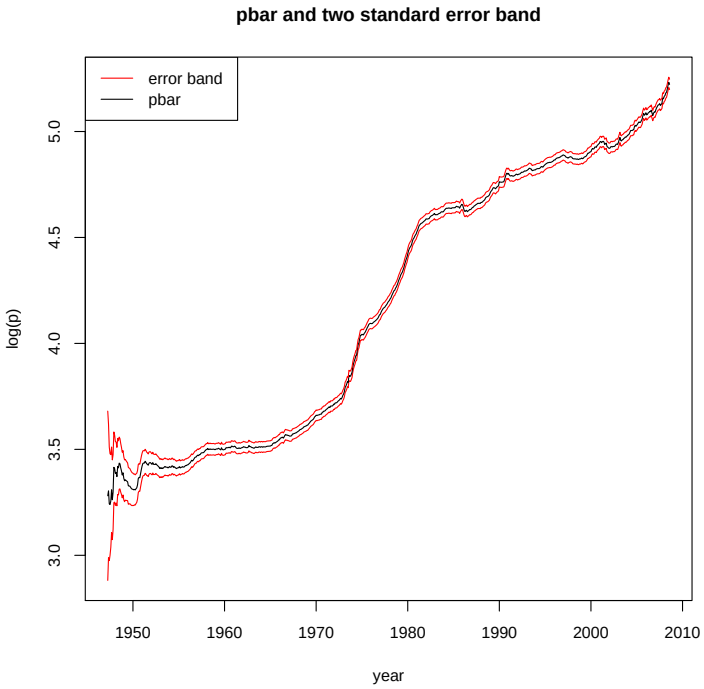
(c) Find $E_t[X_5], t = 1, \ldots, 5$ for ω_3 and ω_5. (These will be two functions of t mapping $\{1, \ldots, 5\} \to \mathbb{R}$.)

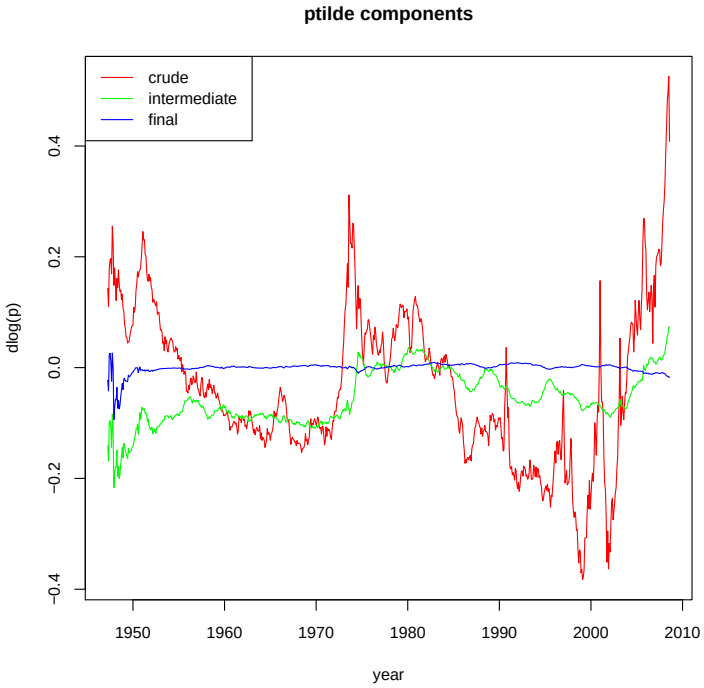
ω_3 :	$7.6, 20, 20, 0, 0$
ω_5 :	$7.6, -2/3, -2/3, 2, 2$

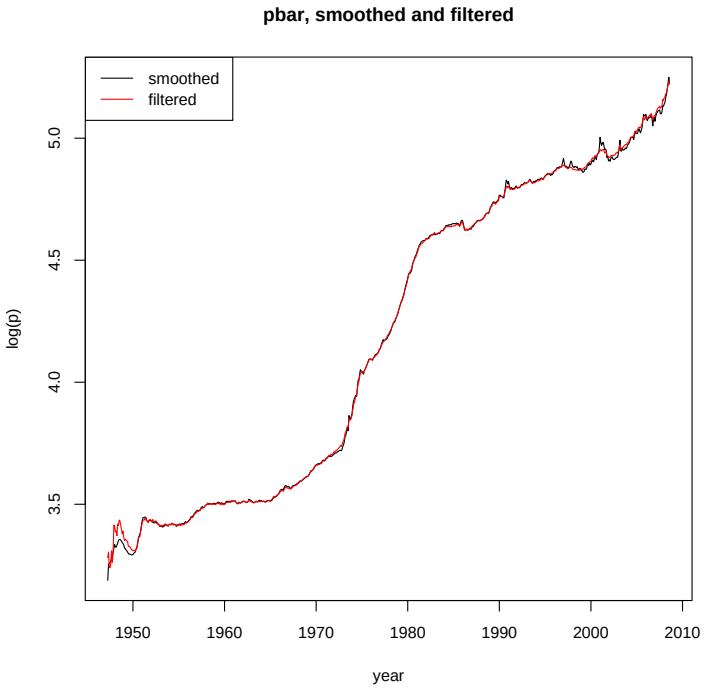
(d) Find $E_t[X_5 \cdot X_4]$, $t = 1, \ldots, 5$ for ω_1 and ω_4.

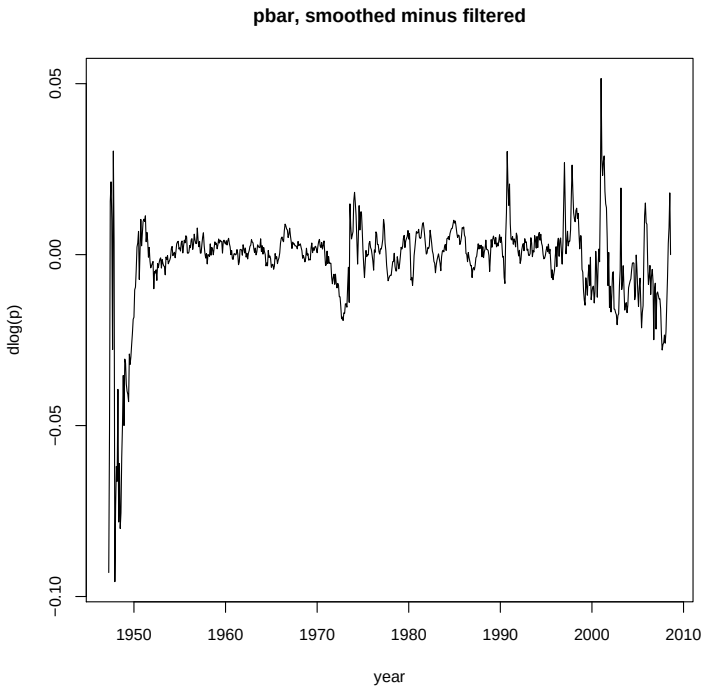
ω_1 :	$11.2, -8, -8, -16$
ω_4 :	$11.2, 40, 40, 80, 80$


3 PPI series and pbar
crude
intermediate
final
pbar
log(p)
3.5
4.0
4.5
5.0
5.5
1950
1960
1970
1980
1990
2000
2010
year


pbar and two standard error band
error band
pbar
log(p)
3.0
3.5
4.0
4.5
5.0
1950
1960
1970
1980
1990
2000
2010
year


ptilde components
crude
intermediate
final
dlog(p)
0.4
0.2
0.0
−0.2
−0.4
1950
1960
1970
1980
1990
2000
2010
year


pbar, smoothed and filtered
smoothed
filtered
log(p)
5.0
4.5
4.0
3.5
1950
1960
1970
1980
1990
2000
2010
year



Kalman filter programs

```
  kfeg0 <- function(param, y, shat0, sig0) {
  ## model is ptil = A(L) ptil +eps
  ##          pbar      = b(L) pbar + c + nu
  ##          pobs      = ptil + pbar
  ##  A and b are polynomials in L and L^2, with A(L) diagonal
  H <- matrix(0,3,9)
  H[ , 1:3] <- diag(3)
  H[ ,4] <- 1
  A <- matrix(0,9,9)
  s <- matrix(param[1:8],4,2)
  param[1:4] <- s %*% c(.5,.5)
  param[5:8] <- s %*% c(.5,-.5)         #use sum and diff of AR coeffs as params, to limit dependence
  rts <- sqrt(param[1:4]^2 + 4 * param[5:8] + 0i) #0i term is to make sqrt handle negative numbers
  rts <- .5 * c(param[1:4] + rts, param[1:4] - rts)
  if (any(abs(rts) > 1.04)) return(list(lh=-1e20))
  penalty <- sum(pmax(abs(rts)-1, 0)^2) * 1e6
  ## with 700 monthly observations, 2% per month explodes too fast
  A[1:4,1:4] <- diag(param[1:4])
  A[1:4, 5:8] <- diag(param[5:8])
  A[4, 9] <- param[9]
  A[5:8, 1:4] <- diag(4)
  A[9,9] <- 1
  M <- matrix(0,9,9)
  M[1:4,1:4] <- diag(exp(param[10:13])) #crossprod(M) is Var(state shocks)
  kfout <- kfeg(y, H, shat0, sig0, G=A, M=M)
  return(c(kfout, list(penalty=penalty)))
}

kfeg <- function(y, H, shat0, sig0, G, M) {
  nT <- dim(y)[1]
```

```
  ny <- dim(y)[2]
  nstate <- length(shat0)
  tvH <- length(dim(H)) > 2
  tvG <- length(dim(G)) > 2
  tvM <- length(dim(M)) > 2
  fixmat <- !(tvH || tvG || tvM)
  if (tvM) stopifnot(c(nstate,nstate, nT) == dim(M)) else stopifnot(c(nstate,nstate) == dim(M))
  if (tvG) stopifnot(c(nstate,nstate, nT) == dim(G)) else stopifnot(c(nstate,nstate) == dim(G))
  if (tvH) stopifnot(c(ny,nstate, nT) == dim(H)) else stopifnot(c(ny, nstate) == dim(H))
  lh <- matrix(0,nT,2)
  fcsterr <- matrix(0,nT,ny)
  shat <- matrix(0,nT, nstate)
  sig <- array(0,c(nT, nstate,nstate))
  Git <- G; Hit <- H; Mit <- M
  for (it in 1:nT) {
    if(!fixmat) {
      if(tvG) Git <- G[ , , it]
      if(tvH) Hit <- H[ , , it]
      if(tvM) Hit <- M[ , , it]
    }
    kfout <- kf2(y[it, ], Hit, shat0, sig0, Git, Mit)
    lh[it, ] <- kfout$lh
    fcsterr[it, ] <- kfout$fcsterr
    sig[it, , ] <- kfout$signew
    shat[it, ] <- kfout$shatnew
    shat0 <- kfout$shatnew
    sig0 <- kfout$signew
  }
  return(list(shat=shat, sig=sig, lh=lh, fcsterr=fcsterr))
}
```

Transcript (with some stupid mistakes and typos removed) of plotting session

```
> kfoutans <- kfeg0(csout$xh, y=ppi3, shat0=c(0,0,0,0,0,0,0,0,1), sig0=sig0)
> str(kfoutans)
List of 5
 $ shat   : num [1:737, 1:9] 0.143 0.110 0.178 0.194 0.197 ...
 $ sig    : num [1:737, 1:9, 1:9] 0.0399 0.0248 0.0180 0.0144 0.0119 ...
 $ lh     : num [1:737, 1:2] -0.433 -0.961 -1.108 -0.452 -0.333 ...
 $ fcsterr: num [1:737, 1:3]  3.42300 -0.41004  0.00603  0.01198  0.01968 ...
 $ penalty: num 0.0248
> plot(cbind(ppi3, kfoutans$shat[,4]), col=c("red","green","blue", "black"),plot.type="single",  main="3 PPI series and
+ pbar", xlab="year", ylab="log(p)")
> legend(x="topleft", legend=c("crude","intermediate","final"),lty=1,col=c("red","green","blue"))
> dev.copy2eps(file="3pAndPbar.eps")
X11cairo
       2
> plot(ts(matrix(kfoutans$shat[,4],737,3) +
+ 2*matrix(c(-sqrt(kfoutans$sig[,4,4]),rep(0,737),sqrt(kfoutans$sig[,4,4])),737,3),start=tsp(ppi3)[1],freq=12),main="pbar
+ and two standard error band", xlab="year", ylab="log(p)",plot.type="single", col=c("red","black","red"))
> legend(x="topleft", legend=c("error band","pbar"),lty=1,col=c("red","black"))
> dev.copy2eps(file="pbarPM2sig.eps")
X11cairo
       2
> plot(ts(kfoutans$shat[,1:3],start=tsp(ppi3)[1],freq=12), col=c("red","green","blue"),main="ptilde
+ components",xlab="year",ylab="dlog(p)",plot.type="single")
> legend(x="topleft", legend=c("crude","intermediate","final"),lty=1,col=c("red","green","blue"))
> dev.copy2eps(file="ptilde.eps")
X11cairo
       2
> smout <- kseg(kfoutans$shat, kfoutans$sig, csout$xh)
> plot(ts(cbind(smout$shatm[,4], kfoutans$shat[,4]), start=tsp(ppi3)[1], freq=12), plot.type="single", col=1:2,
```

```
+ main="pbar, smoothed and filtered", ylab="log(p)", xlab="year")
> legend(x="topleft", legend=c("smoothed","filtered"), col=1:2, lty=1)
> dev.copy2eps(file="pbarSmthFltr.eps")
X11cairo
       2
> plot(ts(smout$shatm[,4] - kfoutans$shat[,4], start=tsp(ppi3)[1], freq=12), plot.type="single", col=1:2, main="pbar,
+ smoothed minus filtered", ylab="dlog(p)", xlab="year")
> dev.copy2eps(file="pbarSmthMinusFltr.eps")
X11cairo
       2
>
```

Kalman smoothing program

```
  kseg <- function(shat, sig, param) {
  A <- matrix(0,9,9)
  s <- matrix(param[1:8],4,2)
  param[1:4] <- s %*% c(.5,.5)
  param[5:8] <- s %*% c(.5,-.5) #use sum and diff of AR coeffs as params, to limit dependence
  A[1:4,1:4] <- diag(param[1:4])
  A[1:4, 5:8] <- diag(param[5:8])
  A[4, 9] <- param[9]
  A[5:8, 1:4] <- diag(4)
  A[9,9] <- 1
  M <- matrix(0,9,9)
  M[1:4,1:4] <- diag(exp(param[10:13])) #crossprod(M) is Var(state shocks)
  omega <- crossprod(M)
  T <- dim(shat)[1]
  shatm <- shat
  sigm <- sig
  for (it in seq(T - 1, 1, by = -1)) {
    smout <- ksmooth(shat[it, ], sig[it, , ], shatm[it+1, ], sigm[it+1, , ], A, omega)
    shatm[it,] <- smout$btT
    sigm[it, , ] <- smout$StT
  }
  return(list(shatm=shatm, sigm=sigm))
}
```