```
clear all
close all
restoredefaultpath
addpath(genpath('jsz_library'))
clc

% This gives a sample estimation of an N-factor "without error" GDTSM.
% See "A New Perspective on Gaussian Dynamic Term Structure Models" by
 Joslin, Singleton and Zhu

% Load some data: mats (1*J) and yields (T*J)
load('sample_zeros.mat')


% Setup format of model/data:
N = 2;
W = pcacov(cov(yields));
W = W(:,1:N)';  % N*J
cP = yields*W'; % T*N
dt = 1/12;

% Estimate the model by ML.
help sample_estimation_fun
VERBOSE = true;
[llks, AcP, BcP, AX, BX, kinfQ, K0P_cP, K1P_cP, sigma_e, K0Q_cP,
 K1Q_cP, rho0_cP, rho1_cP, cP, llkP, llkQ,  K0Q_X, K1Q_X, rho0_X,
 rho1_X] = ...
        sample_estimation_fun(W, yields, mats, dt, VERBOSE);
```

function [llks, AcP, BcP, AX, BX, kinfQ, K0P_cP, K1P_cP, sigma_e, K0Q_cP, K1Q_cP, rho0_cP, rho1_cP, cP, llkP, llkQ, K0Q_X, K1Q_X, rho0_X, rho1_X] = ...
sample_estimation(W, yields, mats, dt, VERBOSE)

Estimates the model with the following setup:

1. Optimize over (lamQ, Sigma_cP).
2. lamQ is assumed to be real, parameterized by the difference to maintain order.
3. Sigma_cP parameterized by cholesky factorization in optimization
4. Take randomized lamQ as initial seeds. See lines 107-108 for the randomization
5. always use OLS estimate of Sigma_cP to start (see Joslin, Singleton, Le)
6. Generate 50 random seeds and take best to start (avoid really bad areas -- no point in looking here)
7. (kinfQ, sigma_e, K0P, K1P) are all concentrated out of the likelihood function. See JSZ and JLS.
8. Run fmincon and repeat 3 times. Repeating re-sets the iteratively computed Hessian.

```
 INPUTS:
 W       : N*J,  weights for the yield portfolios measured without
error
 yields  : T*J,  annualized zero coupon yields
 mats    : 1*J,  maturities, in years
 dt      : scalar, time in years for each period
 VERBOSE : boolean, true prints more output

 OUTPUTS:
 K0Q_X      : N*1,      normalized latent-model matrix
 K1Q_X      : N*N,      normalized latent-model matrix
 Sigma_cP   : N*N,      positive definite matrix that is the
covariance of innovations to cP
 K0P_cP     : N*1,
 K1P_cP     : N*N,
 sigma_e    : scalar,   standard error of yield observation errors
(errors are i.i.d)

 Compute likelihood conditioned on first observation!

 llk        : T*1       time series of -log likelihoods (includes 2-
pi constants)
 AcP        : 1*J       yt = AcP' + BcP'*Xt  (yt is J*1 vector)
 BcP        : N*J       AcP, BcP satisfy internal consistency
condition that AcP*W' = 0, BcP*W' = I_N
 AX         : 1*J       yt = AX' + BX'*Xt
 BX         : N*J       Xt is the 'jordan-normalized' latent state


 The model takes the form:
   r(t) = rho0_cP + rho1_cP'*cPt
        = rinfQ + 1'*Xt  (Xt is the 'jordan-normalized' state
        = 1 period discount rate (annualized)

 Under Q:
   X(t+1) - X(t)   = K0Q_X  + K1Q_X*X(t)  + eps_X(t+1),   cov(eps_X(t
+1)) = Sigma_X
   cP(t+1) - cP(t) = K0Q_cP + K1Q_cP*X(t) + eps_cP(t+1),
cov(eps_cP(t+1)) = Sigma_cP
   where Sigma_X is chosen to match Sigma_cP
 and K0Q_X(m1) = kinfQ where m1 is the multiplicity of the highest
eigenvalue (typically 1)

 Under P:
   cP(t+1) - cP(t) = K0P_cP + K1P_cP*X(t) + eps_cP(t+1),
cov(eps_cP(t+1)) = Sigma_cP

 Model yields are given by:
   yt^m = AcP' + BcP'*cPt  (J*1)
 And observed yields are given by:
  yt^o = yt^m + epsilon_e(t)
 where V*epsilon_e~N(0,sigma_e^2 I_(J-N))
 and V is an (J-N)*J matrix which projects onto the span orthogonal
to the
```

```
 row span of W.  This means errors are orthogonal to cPt and cPt^o =
cPt^m.


Improved seed llk to -28.57
Improved seed llk to -29.027
Improved seed llk to -29.054
Improved seed llk to -29.518
Improved seed llk to -30.279
Improved seed llk to -30.807
Improved seed llk to -31.869
Likelihood on step 1: -35.35524637 parameters:-0.0157 -0.000416
 0.00638 0.00129 0.00227
Likelihood on step 2: -35.35524637 parameters:-0.0157 -0.000416
 0.00638 0.00129 0.00227
Likelihood on step 3: -35.35524637 parameters:-0.0157 -0.000416
 0.00638 0.00129 0.00227
```

Published with MATLAB® R2018b