# Why consider Gibbs sampler?

Less constraints, statistically neater & uncertainty quantification. When is not diagonal, optimization is not convex. (<https://www.pnas.org/content/116/42/20881>).

Maybe useless, just give a try…

# Calculate covariance matrix for all state vectors

The adaptive smoother can only give covariance matrix at each step. But how to calculate covariance matrix for all? This is important when doing Gibbs sampling and Q estimation.

Let

Let , and be the log-likelihood at step k. Then,

So, the gradient is:

And the block tri-diagonal hessian is:

(For robustness, may use Fisher scoring in hessian, as shown in adaptive filtering)

Use the adaptive smoother to get the mean and then calculate the covariance matrix as .

# Gibbs Sampler

Assume . is the dimension of state vectors.

## Conditional Prior

The initial is . Assume is known (e.g. ). The prior for is , where and . The prior for has 2 versions:

1. is diagonal

The diagonal element of is .

Where and

1. No constraints on

Where and .

(To make the mean of loosely centered around )

## MCMC iteration:

1. Update

Use adaptive smoother (with or without window) to get the posterior mean and calculate/estimate the posterior covariance as above.

Then sample from the (approximate) full conditional distribution . To make best use of sparse structure (block tri-diagonal covariance), use Cholesky decomposition of : sample , then . This is super fast.

Unlike Poisson case, the Hessian calculation for CMP is cumbersome. But we can use the approximation shown in <https://www.sciencedirect.com/science/article/abs/pii/S0167947317302608>.

Here I just use truncated summation to show the idea.

1. Update :

By conjugacy,

1. Update :
   1. is diagonal: update

Denote the row of as and element of as . Further, .

By conjugacy,

* 1. No constraints on

Let

By conjugacy,

# Simulation

T = 100, dt = 0.1. code in “demo\Q\_Gibbs\QGibbs\_diag\_demo.m” and “demo\Q\_Gibbs\QGibbs\_NoDiag\_demo.m”.

## Diagonal Q



idx = 200:ng;

mean(Q\_fit(:,:,idx), 3)

% ans =

%

% 1.0e-03 \*

%

% 0.5507 0

% 0 0.0764

## Unconstraint Q



idx = 200:ng;

mean(Q\_fit(:,:,idx), 3)

% ans =

%

% 1.0e-03 \*

%

% 0.8858 0.1452

% 0.1452 0.0598

Even when the ground truth is diagonal (truth = diag([1e-3 1e-5])), the unconstraint version looks better (if there’s no bug). But… both are not perfect: there are lots of autocorrelations.