# Testing dpf

*Daniel J. McDonald*

*10/23/2017*

## Testing

I tried doing the following:

```
# pull recent mtmcbride/dpf/dpf
# devtools::install_github('mtmcbride/dpf/dpf')
library(dpf)
```

which had some problems that I'm not quite sure of. I managed to download it, open the file structure and build using `devtools::load_all()`.

```
##devtools::load_all('../dpf')
## Note that this is my forked version.
```

I then made a few modifications for testing to `dpf.cpp`. These are incorporated in my fork. (Which has the same issue). In `kf1step` I used

```
arma::mat a1 = a0;
arma::mat pred = ct + Zt * a0; // added this between 152 and 153
arma::mat vt = yt - pred;
arma::mat Ft = GGt + Zt * P0 * Zt.t();
arma::mat Ftinv = arma::inv(Ft);
arma::mat Kt = P0 * Zt.t() * Ftinv;
a1 += Kt * vt;
```

and changed the output to

```
return List::create(Named("a1") = a1,
                    Named("P1") = P1,
                    Named("lik") = lik,
                    Named("pred") = pred); // added this line 172
```

These two changes should allow me to see the predictions (the continuous states you actually want to plot) for the *previous* state.

Second, I changed the following in `pathStuff`:

```
means(iter) = step["pred"];
arma::mat aa0 = step["a1"];
arma::mat PP0 = step["P1"];
double liktmp = step["lik"];
llik(iter) += log(liktmp);
```

And made it return a list:

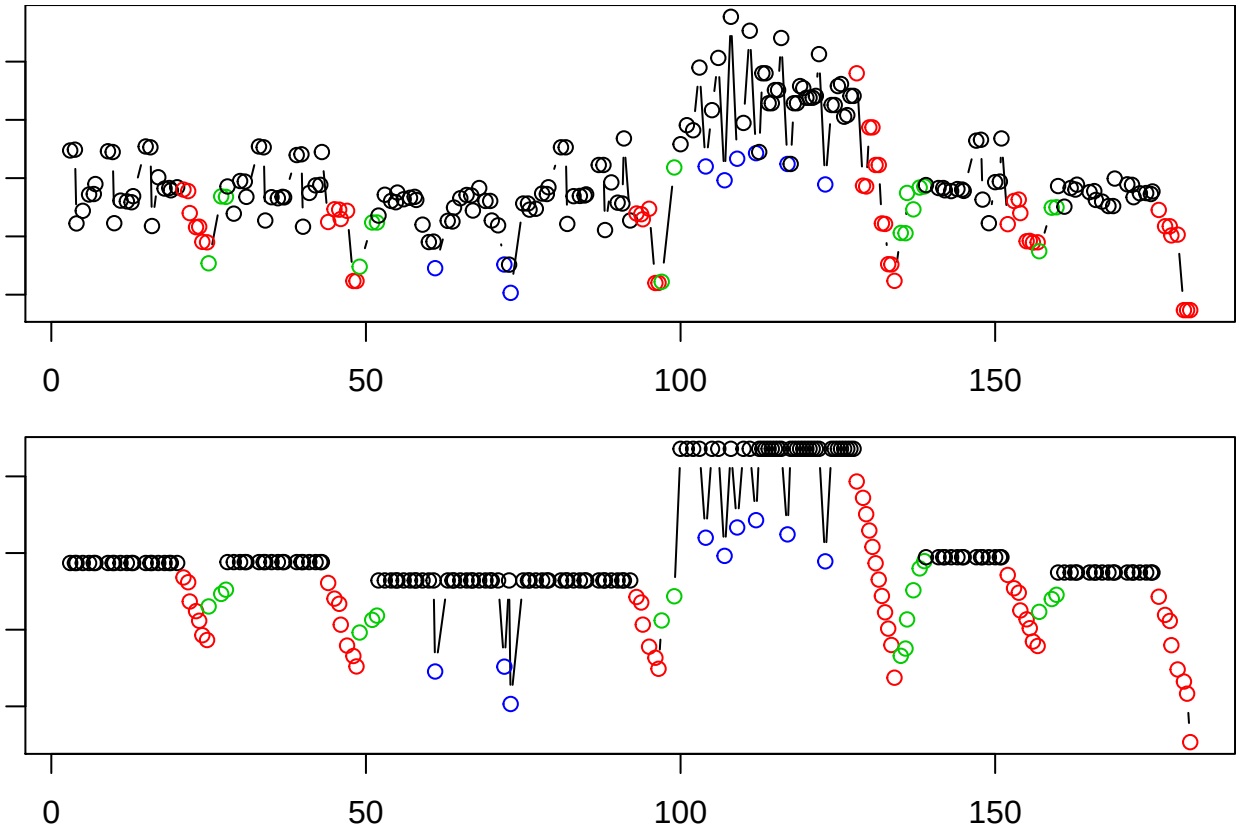
```
return List::create(Named("preds") = means,
                        Named("llik") = llik);
```

This should let me look at the predictions easily if I get a path.

Finally, I removed the `temposwitch` argument from `yupengMats` and reduced the number of $mu$ parameters to 3.

At this point, I get the same issues that you had (all the predictions are the same).

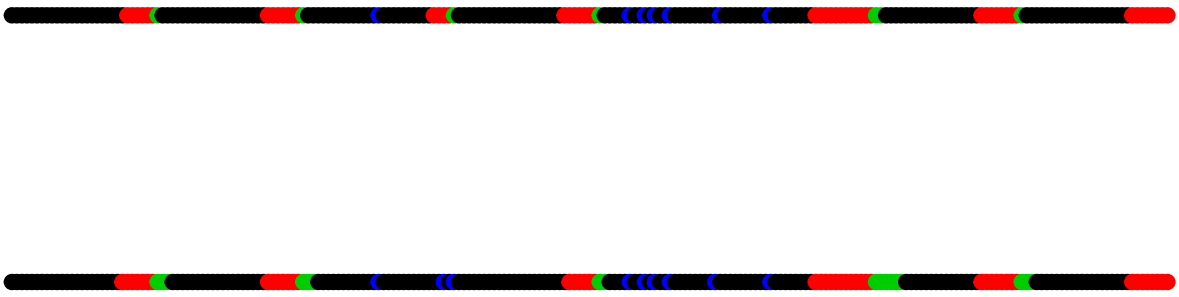
```
load('../music.Rdata')
source('../Beam_Search.R') ## Yupeng's original beam search on some music data
```




This plots our discrete states and his.

```
lt = ioi
out = yupengMats(lt, sd_l^2, c(mean(y),-40,-40), c(.01,20,10,30)^2, c(.8,.1,.8,.4))
test = beamSearch(out$a0,out$P0,c(1,0,0,0,0,0,0,0), out$dt, out$ct, out$Tt, out$Zt,
out$Rt, out$Qt, out$GGt, matrix(y,nrow = 1), out$transMat, 50)
source('../../R/stateConversion.R')
nn = length(y)
bestpath = test$paths[which.max(test$weights),]
par(mfrow=c(1,1))
par(mar=c(0,0,0,0),family='serif')
plot(1:nn,rep(1,nn),col=convert8to4(bestpath),
     pch=19,ylim=c(-1,2),bty='n',xlab='',ylab='',yaxt='n',xaxt='n')
points(1:nn, rep(0,nn), col=s, pch=19)
text(100,1.5,"ours")
text(100,-.5,"yupeng's")
```

ours



yupeng's

This plots our continuous states.

```
filt = pathStuff(out, bestpath, matrix(y,nrow=1))
par(mar=c(5,4,0,0),family='serif')
plot(note.onset,filt$ests,col=convert8to4(bestpath),
     pch=19,bty='n',xlab='note #',ylab='bpm',ylim=range(y))
lines(note.onset, y)
```