Active Subspaces with BASS (Simple Polynomial Examples)

Kellin Rumsey

Simple Polynomial Benchmark

This is the example found in Section 4.1 of the main manuscript. The function of interest is

```
# A simple function of two variables
f <- function(x){
  x[1]^2 + x[1]*x[2] + x[2]^3/9
}
```

Although this is a function of just 2 variables, we will "pretend" that it is a function of $p = 5$ inputs and generate $n = 500$ observations from this function as follows

```
n <- 500
p <- 5
X <- matrix(runif(n*p), ncol=p)
y <- apply(X, 1, f)
```

The true C matrix for this function is given by

```
Ctrue <- matrix(0, nrow=p, ncol=p)
Ctrue[1:2, 1:2] <- matrix(c(120, 50,
                            50, 21) / 45,
                          nrow=2, byrow=TRUE)
round(Ctrue, 2)
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 2.67 1.11    0    0    0
[2,] 1.11 0.47    0    0    0
[3,] 0.00 0.00    0    0    0
```

```
[4,] 0.00 0.00    0    0    0
[5,] 0.00 0.00    0    0    0
```

We can estimate this matrix using the `concordance` package as follows

```r
#devtools::install_github("knrumsey/concordance")
library(concordance)
library(BASS)

mod <- bass(X, y, verbose=FALSE)
C   <- C_bass(mod)
round(C, 3)
```

```
      [,1]  [,2] [,3] [,4] [,5]
[1,] 2.656 1.107    0    0    0
[2,] 1.107 0.466    0    0    0
[3,] 0.000 0.000    0    0    0
[4,] 0.000 0.000    0    0    0
[5,] 0.000 0.000    0    0    0
```

A complete analysis of this function, including a comparison between the `C_bass()` and the `activegp::C_gp()` functions can be found in the script `R/A_simple_polynomial.R`.

A Linear Constraint

As in Section 4.1.1 of the manuscript, we consider the same function as before, but with the prior over inputs x as

$$\rho_2(x_1, x_2, \dots x_p) = 2\mathbb{1}(0 < x_2 < x_1 < 1)\prod_{i=3}^{p}\mathbb{1}(0 < x_i < 1)$$

The true C matrix in this setting can be analytically found as

```r
Ctrue2 <- matrix(0, nrow=p, ncol=p)
Ctrue2[1:2, 1:2] <- matrix(c(1710, 741,
                             741, 322) / 540,
                           nrow=2, byrow=TRUE)
round(Ctrue2, 2)
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 3.17 1.37    0    0    0
[2,] 1.37 0.60    0    0    0
[3,] 0.00 0.00    0    0    0
[4,] 0.00 0.00    0    0    0
[5,] 0.00 0.00    0    0    0
```

To approximate this setting, we use a mixture of uniform priors (see equation (34) in the manuscript for details). For example, using a mixture of $L = 3$ uniform components yields the following priors

```
L <- 15 # Number of mixture components
rho <- list()
for(i in 1:L){
  rho[[i]] <- list(
                rho1=list(dist="uniform", trunc=c(i/(L+1), 1)),
                rho2=list(dist="uniform", trunc=c((i-1), i)/(L+1)),
                rho3=list(dist="uniform", trunc=c(0,1)),
                rho4=list(dist="uniform", trunc=c(0,1)),
                rho5=list(dist="uniform", trunc=c(0,1))
                )
}
```

Next, we find the C matrix corresponding to each of the components. Note that we can use the same BASS model as before. For description of the weights, see the discussion following equation 34 in the manuscript.

```
C_ell <- list()
wt <- rep(NA, L)
for(i in 1:L){
  C_ell[[i]] <- C_bass(mod, prior=rho[[i]])
  wt[i] <- (1 - i/(L+1))*(1/(L+1))
}
wt <- wt/sum(wt)
```

Finally, we can estimate C by aggregating the C for each component.

```
C <- matrix(0, nrow=p, ncol=p)
for(i in 1:L){
  C <- C + C_ell[[i]]*wt[i]
}
round(C, 3)
```

```
      [,1]  [,2] [,3] [,4] [,5]
[1,] 3.146 1.373    0    0    0
[2,] 1.373 0.603    0    0    0
[3,] 0.000 0.000    0    0    0
[4,] 0.000 0.000    0    0    0
[5,] 0.000 0.000    0    0    0
```

Which matches the true answer very closely. For a more complete analysis over various values of L, see the script `R/B_linear_constraint.R`.

Capturing Low-Dimensional Structure

We now turn to the example from Section 4.1.2 in the manuscript. In this example, we define the function g in terms of f by projecting a 6 dimensional input vector into two dimensions before feeding it into f. Training data is generating as

```
n <- 500
p <- 6
X <- matrix(runif(n*p), ncol=p)
A <- matrix(c(12, 9, 6, 1,  1, 1,
              1,  1, 1, 15, 6, 6) / 3,
            nrow=p, ncol=2)
Z <- X%*%A
y <- apply(Z, 1, f)
```

We obtain C for the function g with the `concordance` package as follows

```
mod <- bass(X, y, verbose=FALSE)
C   <- C_bass(mod)
```

The (full 6-dimensional) active subspace can be found as

```
W <- eigen(C)$vectors
```

To reconstruct Table 1 from the manuscript, we can fit new BASS models by using just the first $q \leq p$ *active variables*. For instance, the first 2 active variables contain nearly all of the information needed to build an excellent statistical summary of the function g.

```
q     <- 2
mod_q <- bass(X%*%W[,1:q], y, verbose=FALSE)
rmspe <- sqrt(mean((mod_q$yhat.mean - y)^2))
```

```
# Percent of SD Explained
round((1 - rmspe/sd(y))*100, 3)
```

```
[1] 98.674
```

Code to fully reproduce Table 1 is given in the script ‘R/C_low_dim_structure.R‘.