Global Tutorial for GOALS

Emily Winn-Nunez

2023-10-20

Global Tutorial for GOALS

This script demonstrates how to calculate global GOALS values given a data set for each covariate. Specifically, this file (1) shows how to calculate the Gaussian Kernel, (2) how to precompute before calling the function to calculate GOALS and (3) Calculating the GOALS matrix and the global GOALS themselves.

NOTE: This script is based on a simple (and small) genetics example where we simulate genotype data for n individuals with p measured genetic variants. We randomly assume that three of the predictors j = {23, 24, 25} are causal and have true association with the generated (continuous) phenotype y. We then assume that the j predictive markers explain a fixed H^2% (phenotypic variance explained; PVE) of the total variance in the response $V(y)$. This parameter H^2 can alternatively be described as a factor controlling the signal-to-noise ratio. The parameter ρ represents the proportion of H^2 that is contributed by additive effects versus interaction effects. Namely, the additive effects make up ρ%, while the pairwise interactions (epistatic effects) make up the remaining $(1 - \rho)$%.

```r
### Load in the C++ BAKR functions ###
sourceCpp("C:/Users/etwin/git_repos/GOALS/Software/BAKRGibbs.cpp")
#Register cores
cores = detectCores()
registerDoParallel(cores=cores)
```

Simulation of data

```r
### Set the random seed to reproduce research ###
set.seed(11151990)

n = 2e3; p = 25; pve=0.6; rho=0.5;

### Define the Number of Causal SNPs
ncausal = 3

### Simulate Synthetic Data ###
maf <- 0.05 + 0.45*runif(p)
X   <- (runif(n*p) < maf) + (runif(n*p) < maf)
X   <- matrix(as.double(X),n,p,byrow = TRUE)
Xmean=apply(X, 2, mean); Xsd=apply(X, 2, sd); Xc=t((t(X)-Xmean)/Xsd)
s=c(23:25)

#Marginal Effects Only
Xmarginal=Xc[,s]
```

```
beta1=rep(1,ncausal)
y_marginal=c(Xmarginal%*%beta1)
beta1=beta1*sqrt(pve*rho/var(y_marginal))
y_marginal=Xmarginal%*%beta1

#Pairwise Epistatic Effects
Xepi=cbind(Xc[,s[1]]*Xc[,s[3]],Xc[,s[2]]*Xc[,s[3]])
beta2=c(1,1)
y_epi=c(Xepi%*%beta2)
beta2=beta2*sqrt(pve*(1-rho)/var(y_epi))
y_epi=Xepi%*%beta2

#Error Terms
y_err=rnorm(n)
y_err=y_err*sqrt((1-pve)/var(y_err))

y=c(y_marginal+y_epi+y_err); #Full Model
colnames(X) = paste("SNP",1:ncol(X),sep="")
```

Create the Nonlinear Covariance Matrix with the Gaussian Kernel

This function takes on two arguments: (1) The Genotype matrix X. This matrix should be fed in as a $p \times n$ matrix. That is, predictor are the rows and subjects/patients/cell lines are the columns. (2) The bandwidth (also known as a smoothing parameter or lengthscale) h. For example, the Gaussian kernel can be specified as $k(u, v) = \exp ||u - v||^2/2h^2$. Default bandwidth is $h = 1$.

```
n = dim(X)[1] #Sample size
p = dim(X)[2] #Number of markers or genes
```

Run the GOALS function

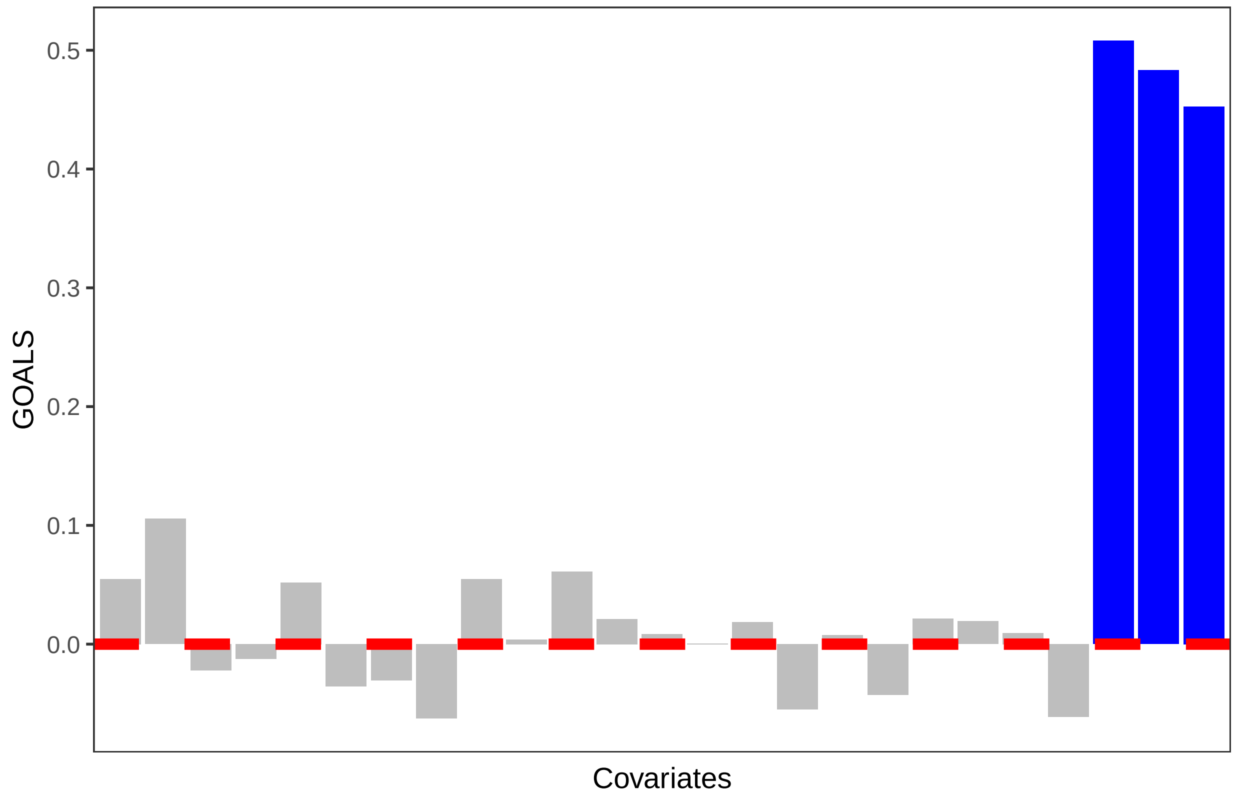
Find the Approximate Basis and Kernel Matrix; Choose $N \leq D \leq P$. Complete all pre-calcalculation for GOALS function.

```
sigma2 = 1e-2
B = GaussKernel(t(X)); diag(B)=1
A <- B + sigma2*diag(1,nrow=n,ncol=n)
A_svd <- svd(A)
Ainv = nearPD(A_svd$v%*%diag(1/A_svd$d, nrow=n, ncol=n)%*%t(A_svd$u))$mat
Aiy <- Ainv%*%y
BAiy <- B %*% Aiy
IAiB <- (diag(1,nrow=n, ncol=n)-Ainv%*%B)
BIAiB <- B%*%IAiB

### Compute GOALS matrix ###
delta = ComputeESAFast(X, B, as.vector(Aiy), as.vector(BAiy), cores=cores)

### Get Global GOALS scores by averaging over columns ###
global_delta = colMeans(delta)
```

Plot the GOALS values

Color coding associates gray with noncausal SNPs and blue with causal SNPs.

```r
snp_names = paste("SNP", 1:p, sep="")
my_df = data.frame("SNPS"=snp_names, "GOALS"=global_delta)
ggplot(data=my_df, aes(x=SNPS, y=GOALS))+
  geom_bar(stat="identity", fill = c(rep("gray", 22), rep("blue",3)))+
  scale_x_discrete(limits=snp_names) +
  ggtitle("")+
  geom_hline(yintercept=0,linetype="dashed", color="red",linewidth=2)+
  labs(x="Covariates")+
  theme_bw()+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        legend.position="none", axis.text.x=element_blank(), axis.ticks.x=element_blank())
```


GOALS
Covariates