Ridge Regression: problem

- Originally proposed as a regularization procedure for linear regression in case of multi-collinearity

$$\text{minimize } \sum_p (E_p - \sum_m N_{pm} A_m)^2 \text{ subject to } \sum_m A_m^2 \leq c$$
$$\Leftrightarrow \quad \text{minimize} \sum_p (E_p - \sum_m N_{pm} A_m)^2 + \lambda \sum_m A_m^2$$

- Bayesian interpretation:

$$P(A|\lambda, \sigma) \propto \prod_m \exp\left[-\frac{\lambda}{2\sigma} A_m^2\right]$$
$$P(E|N, A, \lambda, \sigma) \propto \prod_p \exp\left[-\frac{(E_p - \sum_m N_{pm} A_m)^2}{2\sigma}\right]$$
$$\Leftrightarrow P(A|N, E, \lambda, \sigma) \propto P(E|N, A, \lambda, \sigma) P(A|\lambda, \sigma)$$

Ridge Regression: solution

$$P(A|N,E,\lambda,\sigma) \propto P(E|N,A,\lambda,\sigma)P(A|\lambda,\sigma)$$

Solution to the above stated problem:

$$\mathrm{Mean}(A_m) = \sum_{\bar{m}} C_{m\bar{m}} \sum_p N_{p\bar{m}} E_p$$

$$\mathrm{Var}(A_m) \propto C_{mm}$$

where $C_{m\bar{m}} = (\sum_P N_{pm} N_{p\bar{m}} + \delta_{m\bar{m}}\lambda)^{-1}_{m\bar{m}}$.

So what do we need to do?

- Efficient computation of matrix inverse and inner products
- (Generalized) cross-validation to fit λ

Ridge Regression: computation

Given the SVD of the site count matrix $N = UDV^T$

$$\text{Mean}(A_m) = \sum_{\bar{m}} V_{m\bar{m}} \frac{D_{\bar{m}\bar{m}}}{D^2_{\bar{m}\bar{m}} + \lambda} \sum_p U_{p\bar{m}} E_p$$

$$\text{Var}(A_m) \propto \sum_{\bar{m}} V_{m\bar{m}} \frac{1}{D^2_{\bar{m}\bar{m}} + \lambda} V_{m\bar{m}}$$

Computation of the SVD of matrix $N^{(P\times M)}$ takes $O(PM^2)$.
Because $P \gg M$ it is more efficient to compute

$$SVD(N^T N) = VD^2V^T, \quad \sim O(M^3)$$

$$\rightarrow U = NV\frac{1}{\sqrt{D}}$$

Ridge Regression in R using SVD

```
ridge.regression = function(N,E,lambda) {

  Ns = fast.svd(N) # a list with entries: u, d, v
  rhs = crossprod(Ns$u,E)
  dia = Ns$d/(Ns$d^2 + nrow(N)*lambda)
  Ahat = sweep(Ns$v,2,dia,FUN='*') %*% rhs

  dimnames(Ahat) = list(colnames(N),colnames(E))

  Chi2 = colSums((E - N %*% Ahat)^2)
  fov = 1-Chi2/colSums(E^2)

  C =  tcrossprod(sweep(Ns$v,2,1/(Ns$d^2 + nrow(N)*lambda),FUN='*'),Ns$v)
  AhatSE = sqrt(diag(C) %x% t(Chi2/nrow(E)))
  Zscore = Ahat/AhatSE
  combined.Zscore = sqrt(rowMeans(Zscore^2))

  fit = list(Ahat=Ahat,Zscore=Zscore,combined.Zscore=combined.Zscore,fov=fov)

  return(fit)
}
```

Ridge Regression: fitting λ

- K-fold cross-validation is known to over-estimate the CV-error leading to conservative λ estimates
- For all models of the form $\widehat{E} = HE$ there is a exact solution to the leave-one-out cross validation error

$$\sum_p (E_p - \sum_m N_{pm} A_m^{-p})^2 = \sum_p \left(\frac{E_p - \sum_m N_{pm} A_m}{1 - H_{pp}} \right)^2$$

where A_m^{-p} is the activity estimate leaving out the p-th data point

- Generalized cross-validation approximates this by

$$\text{GCV-error} \approx \sum_p \left(\frac{E_p - \sum_m N_{pm} A_m}{1 - \sum_p H_{pp}/P} \right)^2$$

- Using the SVD of N, $H = U \frac{D^2}{D^2+\lambda} U^t$, so its easy to compute the trace of H.

Ridge Regression in R: fitting λ

```
optimize.lambda = function(N,E) {

  Ns = fast.svd(N)
  rhs = crossprod(Ns$u,E)

  lambda.bnd = 10^c(-12,-6) * nrow(N) * ncol(N)

  gcv.error = function(lambda,E,Ns,rhs) {
    D = Ns$d^2/(Ns$d^2 + nrow(N)*lambda) # Hat matrix: H = UDU^t
    resid = E - sweep(Ns$u,2,D,FUN='*') %*% rhs
    GCV = sum((resid/(nrow(E)-sum(D)))^2)
    return(GCV)
  }

  opt = optimize(gcv.error,lambda.bnd,E,Ns,rhs)
  lambda.opt = opt$minimum
  gcv.opt = opt$objective

  return(list(lambda.opt = lambda.opt,
              gcv.opt = gcv.opt))
}
```

Huvec time course example

```
> source('ridgeInR.R')
> load('huvec_example.RData')
> Ns = center.cols(N)
> Es = center.cols(E)
> Es = center.rows(Es)
> opt = optimize.lambda(Ns,Es)
> opt
$lambda.opt
[1] 0.004151515

$gcv.opt
[1] 9.378605e-05

> r = ridge.regression(Ns,Es,opt$lambda.opt)
> mean(r$fov)
[1] 0.03283837
> sort(r$combined.Zscore,decreasing=TRUE)[1:10]
      IRF1.2.7.p3 NFKB1_REL_RELA.p2           XBP1.p3          PRDM1.p3
        11.840889          6.896939          3.911662          3.439450
       E2F1..5.p2      NFY.A.B.C..p2           HIC1.p2     FOX.D1.D2..p2
         3.115655          2.581088          2.565535          2.486906
          GUGCAAA  HNF4A_NR2F1.2.p2
         2.473904          2.454952
```

You will get the results of the web page if you use $\lambda = 0.0113087$ (determined by a 5-fold CV run) instead.