MA 589 — Computational Statistics

Project 1

(Due: Thursday, 09/29/16)

1. (WaRming up) Write (`R`) functions that return:

   (a) The identity matrix of order n, I_n (call this function `eye`.) If the input is a matrix `X`, return an identity matrix of order `ncol(X)`[1].

   (b) The Hilbert matrix of order n, H_n (call this function `hilbert`.) H_n has entries $(H_n)_{ij} = (i + j - 1)^{-1}$, for $i, j = 1, \ldots, n$.

   (c) The trace `tr` of a matrix.

   (d) The L_2 norm of vector v with n entries, defined as $\texttt{norm2}(v) = \sqrt{\sum_{i=1}^{n} v_i^2} = \sqrt{v^T v}$. Quick check: what is `norm2(1e200 * rep(1, 100))`?[2]

   (e) The inverse of $U^T U$, if U is an upper triangular matrix of full rank (call this function `invtri`.) (Hint: use `eye` and `backsolve`.)

2. The *machine epsilon*, ϵ, can be defined as the smallest floating point (with base 2) such that $1 + \epsilon > 1$, that is, $1 + \epsilon/2 == 1$ in machine precision.

   (a) Write a function that returns this value by starting at `eps = 1` and iteratively dividing by 2 until the definition is satisfied.

   (b) Write a function that computes $f(x) = \log(1 + \exp(x))$ and then evaluate: $f(0)$, $f(-80)$, $f(80)$, and $f(800)$.

   (c) How would you specify your function to avoid computations if $x \ll 0$ ($x < 0$ and $|x|$ is large)? (Hint: ϵ.)

   (d) How would you implement your function to not overflow if $x \gg 0$?

3. The number of ways we can partition a set of n elements, $n \geq 1$, into k non-empty subsets, $1 \leq k \leq n$, is called the *Stirling number of the second kind* $S(n, k)$. For instance, it can be shown, using combinatorial arguments, that $S(n, 1) = S(n, n) = 1$ and that $S(n, n-1) = n(n-1)/2$. These numbers can be calculated using the formula:

$$S(n,k) = \frac{1}{k!} \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} j^n.$$

   (a) Write a function that computes $\log S(n, k)$, called `lstirling`. Here are the values for $S(10, k)$, $k = 1, \ldots, 10$: 1, 511, 9330, 34105, 42525, 22827, 5880, 750, 45, 1; [3] check that your routine gets them right.

[1] You guessed right: the number of columns of `X`.

[2] `1e201`, of course.

[3] According to `https://oeis.org/A008277`.

(b) Simple check: what is $\log S(100, 99)$ and $\log S(100, 100)$ according to your routine? What went wrong in either case?

(c) $*$ A better way of computing $\log S(n, k)$ is to use a recursion based on the recurrence relation $S(n, k) = kS(n-1, k) + S(n-1, k-1)$. Write a new `lstirling` based on this relation and perform the checks above.

4. (Ridge regression) Let us now practice using the QR decomposition to find least-squares solutions to linear systems. Define $X =$ `hilbert(7)` and $y =$ `rep(1, 7)`.

(a) Use the QR decomposition of X to find the least-squares solution $\hat{\beta}$ that minimizes

$$SS(\beta) = (y - X\beta)^T(y - X\beta). \tag{1}$$

What is the L_2 norm of $\hat{\beta}$?

The matrix X is "poorly conditioned", that is, small perturbations in y can yield large perturbations in $\hat{\beta}$. To alleviate the problem we can penalize the norm of $\hat{\beta}$ by minimizing the following *regularized* sum of squares:

$$SS_\lambda(\beta) = (y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta,$$

where $\lambda > 0$.

Fortunately, minimizing SS_λ is the same as minimizing SS in Equation (1) but with extra "artificial" data,

$$SS_\lambda(\beta) = \left(\begin{bmatrix} y \\ 0_7 \end{bmatrix} - \begin{bmatrix} X \\ \sqrt{\lambda} I_7 \end{bmatrix} \beta \right)^T \left(\begin{bmatrix} y \\ 0_7 \end{bmatrix} - \begin{bmatrix} X \\ \sqrt{\lambda} I_7 \end{bmatrix} \beta \right).$$

(Note that 0_7 is `rep(0, 7)` and I_7 is `eye`(X).) The least-squares solution $\hat{\beta}_\lambda$ is a function of the penalty λ [4].

(b) For `l = 0:17` and $\lambda =$ `10^(-l)`, plot `l` against the L_2 norms of the least-squares solutions $\hat{\beta}_\lambda$. (Hint: check `rbind` to extend X, and use `qr` and its related functions.) What happens as you increase the regularization?

The *effective degrees of freedom* is defined as the trace of S_λ,

$$\texttt{tr}(S_\lambda) = \texttt{tr}\big(X(X^TX + \lambda I)^{-1}X^T\big) = \texttt{tr}\big((I + \lambda(R_1^TR_1)^{-1})^{-1}\big),$$

where R_1 is the upper triangular submatrix in R from the QR decomposition of X [5].

(c) Write a function `effdf` that computes the effective degrees of freedom as a function of λ. (Hint: use `invtri` and note that $I + \lambda(R_1^TR_1)^{-1}$ is positive-definite.)

(d) Now plot `l = 0:17` against `effdf`(λ) for $\lambda =$ `10^(-l)`. Describe what happens as you increase the regularization.

[4] We'll discuss how to pick a good value for λ when we talk about *cross-validation*.

[5] Can you show this?