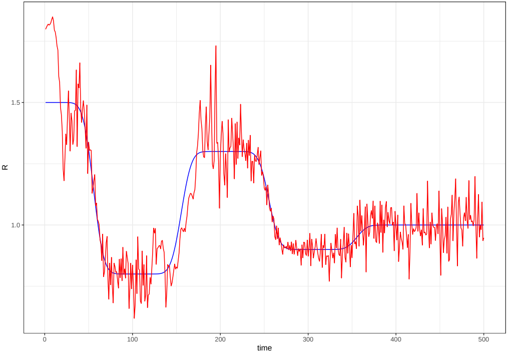
Nov 12 Update Report

2022-11-13

Last Time

- Showed and compared results of 5 methods
- Updates on penalized least squares

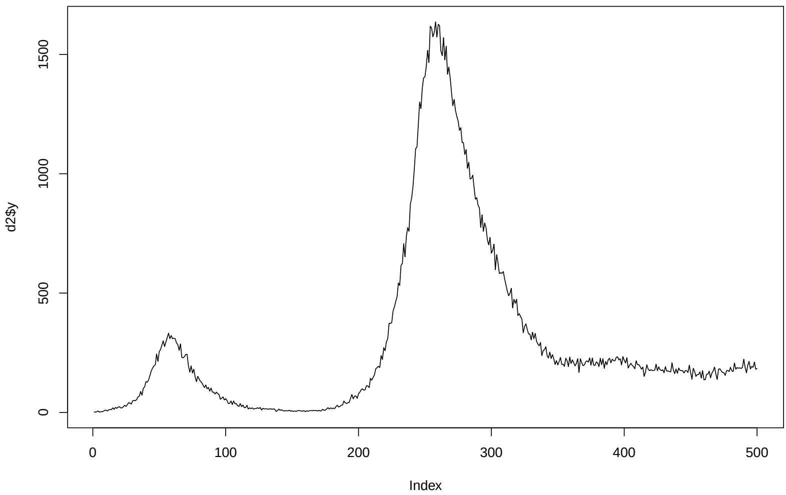
Performance of Ridge regression



"Optimal" lambda chosen is

```
## [1] 403.4288
```

simulation



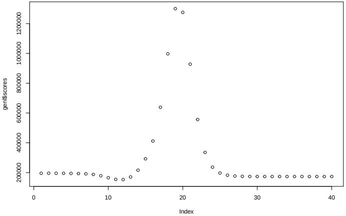
Choice of penalty

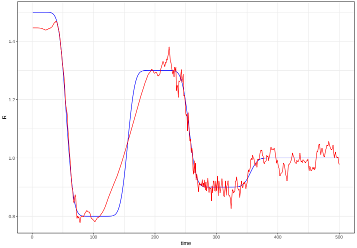
- How I choose lambda
  - Create a grid of lambda, $\lambda \in exp(1:10)$
  - For each λ, calculate the loss
- Why not use cross validation?
  - Assume data $X \in \mathcal{R}^n$, the predictors is of the form $\beta \in \mathcal{R}$. In our case, $\beta \in \mathcal{R}^n$, i.e. sequential
  - If we have a leave out set $k_i = (x_i, y_i)$, and train the model with $k_{-i} = (x_{-i}, y_{-i})$, and get β_{-i}. How to calculate $\hat{y}_i$ given x_i and β_{-i}

k-fold Cross validation from Genlasso

- k = 3
- Split data into c(x, 1, 2, 3, 1, 2, 3,. . . , 2, 3, x), omit the 1st and last items
- Hold out 1, and train on group 2, 3 and get predictor (in our case, r) value. Then interpolate predictor value at position 1. Predict the response (y) with the interpolated predictor and the hold out explanatory (w)
- Repeat and hold out 2 and 3

Result: Scores





Optimal lambda is:

```
## [1] 162754.8
```

Penalizing smoothness using L1-norm

- Setup:
  - Let y_t be the daily case count at day t
  - Then $y_t \sim Pois(r_t * w_t)$, where $w_t = \sum_a y_{t-a} w_a$
- Objective function:
  - $argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \lambda ||Dr||_1$
- Scaled Augmented Lagrangian:
  - Let $Dr = z$, adding penalty for being not equal
  - $L(r, u, z) = \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \lambda ||z||_1 + \frac{\rho}{2} ||Dr - z + u||_2^2 + \frac{\rho}{2} ||u||_2^2$

Continued

- $L(r, u, z) = \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \lambda||z||_1 + \frac{\rho}{2}||Dr - z + u||_2^2 + \frac{\rho}{2}||u||_2^2$
- We could start optimization here:
  - $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \frac{\rho}{2}||Dr - z + u||_2^2$
  - $z \leftarrow argmin_z \lambda||z||_1 + \rho||Dr - z + u||_2^2$
  - $u \leftarrow u + Dr - z$

Further simplification

- Notice that solving $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \frac{\rho}{2}||Dr - z + u||_2^2$ requires matrix inversion
  - Linearize the quadratic term (by Neal Parikh, Proximal Algorithms)
- No enforcement on r being positive
  - Instead, penalize $log(r)$
  - ˆ only penalize the log differences, still not enforcing r to be positive
  - (this is weird, should treat r as normally would, but $\sum_{i=1} w_i r_i - y_i log(w_i r_i))$ is $\sum_{i=1} -w_i exp(r_i) + y_i log(w_i exp(r_i)))$ instead)

Linearize r update

- Linearize quadratic term f as $(r - r^o)^T f'(r^o) + \frac{\mu}{2}||r - r^o||_2^2$
- In our case, $f = \frac{\rho}{2}||Dr - z + u||_2^2$
- Quadratic term becomes $\rho r^T(D^T Dr^o - D^T z + D^T u) + \frac{\mu}{2}||r - r^o||_2^2$
- Then the optimization step for r becomes:
- $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} -w_i r_i^o + y_i log(w_i r_i^o)) + \rho r^T(D^T Dr^o - D^T z + D^T u) + \frac{\mu}{2}||r - r^o||_2^2$

Linearize $log(r)$

- Quadratic term becomes $f = \frac{\rho}{2}||Dlog(r) - z + u||_2^2$
- $= \frac{\rho}{2}[(Dlog(r) - z) + u]^2$
- $= \frac{\rho}{2}[(Dlog(r) - z)^2 + 2(Dlog(r) - z)u + u^2]$
- $= \frac{\rho}{2}[l^T D^T D l - 2l^T D^T z + z^2 + 2(Dl - z)u + u^2]$, taking $l = log(r)$
- $\frac{df(r^o)}{dr} = \frac{df(r^o)}{dl} * \frac{dl}{dr} = \rho(D^T Dlog(r^o) - D^T z + D^T u)(r^o)^{-1}$
- Then the optimization step for r becomes:
- $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \rho r^T(D^T Dr - D^T z + D^T u)(r^o)^{-1} + \frac{\mu}{2}||r - r^o||_2^2$

Summary of progress

Now we have choose

- $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i^o - y_i log(w_i r_i^o)) + \rho r^T(D^T D r^o - D^T z + D^T u) + \frac{\mu}{2}||r - r^o||_2^2$
- $r \leftarrow argmin_r \frac{1}{n}(\sum_{i=1} w_i r_i - y_i log(w_i r_i)) + \rho r^T(D^T D r - D^T z + D^T u)(r^o)^{-1} + \frac{\mu}{2}||r - r^o||_2^2$

Continue

- KKT stationarity condition: $\nabla_r L(r,u,z)|_{r=r^o} = 0$

$$\frac{dL(r,u,z)}{dr} = w_i - y_i \frac{w_i}{w_i r_i^o} + \rho r^t (D^T Dlog(r^o) - D^T z + D^T u)(r^o)^{-1} + \mu(r - r^o) = 0$$

$$=> \mu r = -w_i + y_i \frac{w_i}{w_i r_i^o} - \rho r^t (D^T Dlog(r^o) - D^T z + D^T u)(r^o)^{-1}) + \mu r^o$$

- This is the update step of r

Next step

- $\sum_{i=1} w_i r_i - y_i log(w_i r_i))$ is $\sum_{i=1} -w_i exp(r_i) + y_i log(w_i exp(r_i)))$, and penalize Dr only (or $Dexp(r)$?)
- If the above update for r is correct, implement it

Other types of Cross validation

A Cross Validation framework for Signal Denoising with Applications to Trend Filtering, Dyadic CART and Beyond

By Anamitra Chaudhuri and Sabyasachi Chatterjee

Bayesian linear regression for trend filtering