Stepwise regression and friends

Dean Foster

Amazon
Stepwise regression gets “no respect”

- Quite commonly used, but not often studied
- Most statisticians think of it as “evil” or at best useful only to “lazy” scientists
Stepwise regression gets “no respect”

- Quite commonly used, but not often studied
- Most statisticians think of it as “evil” or at best useful only to “lazy” scientists
- But I’m a fan
- This talk will review some of the theoretical results that are known about it
- I’ll give some examples of its value in applied problems
Problem statement: As a scientist

- Goal: predict $Y$
- Inputs: you have millions of $X$’s that can be used to predict $Y$
- Most $X$’s are garbage
- How do you find a small subset of $X$’s that will predict $Y$ well?
15 years ago Bob Stine and I ran a “little” regression (JASA 2004)

- 70,000 features
- 2 million rows
- $Y =$ credit card holder going bankrupt next month
15 years ago Bob Stine and I ran a “little” regression (JASA 2004)
  - 70,000 features
  - 2 million rows
  - $Y =$ credit card holder going bankrupt next month

At the time it caused jaws to drop
15 years ago Bob Stine and I ran a “little” regression (JASA 2004)
- 70,000 features
- 2 million rows
- $Y =$ credit card holder going bankrupt next month

At the time it caused jaws to drop

Tricks:
- Linear model instead of logistic regression (Fast!)
- Dummy variables for interactions (contain signal)
- Interactions (non-linear structure)
- Bennett’s bound to calculate p-values (avoiding over-fitting)
- Stepwise regression!
Problem statement: example
Problem statement: As a mathematician

- **Model:**
  \[ Y_i \sim X_i^\top \beta + \sigma Z_i \]

- **Penalized regression:**
  \[
  \hat{\beta}_\Pi \equiv \arg \min_{\hat{\beta}} \sum_{i=1}^{n} (Y_i - X_i^\top \hat{\beta})^2 + \Pi \sigma^2 |\hat{\beta}|_0
  \]

- $|\hat{\beta}|_0$ is the number of non-zeros in $\beta$
Problem statement: As a mathematician

- **Model:**
  \[ Y_i \sim X_i^\top \beta + \sigma Z_i \]

- **Penalized regression:**
  \[
  \hat{\beta}_\Pi \equiv \arg \min_{\hat{\beta}} \sum_{i=1}^{n} (Y_i - X_i^\top \hat{\beta})^2 + \Pi \sigma^2 |\hat{\beta}|_0
  \]

- $|\hat{\beta}|_0$ is the number of non-zeros in $\beta$
- Non-convex problem
- **Note:** $L1$ is the convex relaxation of $L0$, which leads to Lasso.
Why we care

- Error larger by $p/q$ if we don’t do variable selection
- Huge improvement in accuracy is possible
- Precisely:

$$E(\mu_{Y|X} - \hat{Y}_p)^2 = \frac{p}{q} E(\mu_{Y|X} - \hat{Y}_q)^2$$

- $\hat{Y}_p$ is best fit using all the variables
- $\hat{Y}_q$ is best fit using only the $q$ correct variables
- But, can we find the right subset?
First algorithm

- Try all subsets to find best fitting subset
  - Oops: Slow, and it will say use all the variables
Second algorithm

- Try all subsets and penalize by Bonferroni
  - $|t| > \sqrt{2 \log(p)}$
  - Yes, it is painfully slow. But does it at least find the right subset?
Theorem (F. and George 1994, Donoho and Johnstone 1994)

For any orthogonal $X$ matrix, if $\Pi = 2 \log(p)$, then the risk of $\hat{\beta}_\Pi$ is within a $2 \log(p)$ factor of the target.
Theorem (F. and George 1994, Donoho and Johnstone 1994)

For any orthogonal $X$ matrix, if $\Pi = 2 \log(p)$, then the risk of $\hat{\beta}_\Pi$ is within a $2 \log(p)$ factor of the target.

- The bound is tight.
- (The same bound works for Lasso.)
Theorem (F. and George 1994, Donoho and Johnstone 1994)

For any orthogonal $X$ matrix, if $\Pi = 2 \log(p)$, then the risk of $\hat{\beta}_\Pi$ is within a $4 \log(p)$ factor of the target.
Theorem (F. and George 1994, Donoho and Johnstone 1994)

For any orthogonal $X$ matrix, if $\Pi = 2 \log(p)$, then the risk of $\hat{\beta}_\Pi$ is within a $4 \log(p)$ factor of the target.

- This bound is also tight
- (Lasso is a disaster in this case.)
Theorem (F. and George 1994, Donoho and Johnstone 1994)

For any orthogonal $X$ matrix, if $\Pi = 2 \log(p)$, then the risk of $\hat{\beta}_\Pi$ is within a $4 \log(p)$ factor of the target.

- So finding the right subset of variables can generate a huge win
instead of exhaustive search, we can use search
Greedy runs fast
Called stepwise regression in statistics
How well does it perform?
Greedy = Stepwise regression

- instead of exhaustive search, we can use search
- Greedy runs fast
- Called stepwise regression in statistics
- How well does it perform?
- For orthogonal problems, it works perfectly
- For many $X$’s it will work well.
- But, . . .
### Nasty example for stepwise

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>\ldots</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\ldots</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>\ldots</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>\ldots</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>1</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
<td>1</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
</tbody>
</table>
### Nasty example for stepwise

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>...</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>$-1 + \delta$</td>
<td>$+1 + \delta$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>$+1 + \delta$</td>
<td>$-1 + \delta$</td>
</tr>
</tbody>
</table>

**“Model:”**

$$Y \sim D1 + D2 + \cdots + X1 + X2$$
Nasty example for stepwise

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>...</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>-1 + δ</td>
<td>1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>1 + δ</td>
<td>-1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>-1 + δ</td>
<td>1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>1 + δ</td>
<td>-1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>-1 + δ</td>
<td>1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>1 + δ</td>
<td>-1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>-1 + δ</td>
<td>1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>1 + δ</td>
<td>-1 + δ</td>
</tr>
</tbody>
</table>

... ... ... ... ... ... ... ...

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>...</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>-1 + δ</td>
<td>1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>1 + δ</td>
<td>-1 + δ</td>
</tr>
</tbody>
</table>

Actually:

\[ Y = \frac{(X1 + X2)}{\delta} \]
Nasty example for stepwise

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>...</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
</tbody>
</table>

- Stepwise regression will crash and burn
- It adds all the other inputs before adding either X1 or X2.
Nasty example for stepwise

<table>
<thead>
<tr>
<th>Y</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>...</th>
<th>Dn/2</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>−1 + δ</td>
<td>+1 + δ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
<td>+1 + δ</td>
<td>−1 + δ</td>
</tr>
</tbody>
</table>

Lasso will also crash and burn
Several algorithms have been proposed to solve these

In some cases they run well, in some cases they are a disaster

Fun mathematics—but not really informative as to what to do in practice

A success for stepwise regression

Theorem (Natarajan 1995)
Stepwise regression will have a prediction accuracy of at most twice optimal using at most \( \approx 18 |X| \) variables.

This result was only recently noticed to be about stepwise regression. He didn’t use that term.

The risk inflation is a disaster.

The \( |X| \) is a measure of co-linearity.

This bound can be arbitrarily large.

Brings up two points: we are willing to “cheat” on both accuracy and number of variables. But hopefully not by very much.

L0 regression is hard

Theorem (Zhang, Wainwright, Jordan 2014)
No algorithm exists which achieves all three of the following goals:
- Runs efficiently (i.e. in polynomial time)
- Runs accurately (i.e. risk inflation < \( p \))
- Returns sparse answer (i.e. \( |\hat{\beta}|_0 \ll p \))

Strongest version requires an assumption about complexity (which I can’t understand).

The proof relies on “interactive proof theory” (which I also can’t understand).

The sparsity results depend on the assumptions used. We can get \( |\hat{\beta}|_0 < cq \) easily, and \( |\hat{\beta}|_0 < p^{\alpha} \) with difficulty.

Difficult to improve to \( |\hat{\beta}|_0 \leq p \) since then all the heavy lifting is being done by the accuracy claims.

L0 regression is VERY hard

Theorem (Foster, Karloff, Thaler 2014)
No algorithm exists which achieves all three of the following goals:
- Runs efficiently (i.e. in polynomial time)
- Runs accurately (i.e. risk inflation < \( p \))
- Returns sparse answer (i.e. \( |\hat{\beta}|_0 \ll p \))

Strongest version requires an assumption about complexity (which I can’t understand).

The proof relies on “interactive proof theory” (which I also can’t understand).

The sparsity results depend on the assumptions used. We can get \( |\hat{\beta}|_0 < cq \) easily, and \( |\hat{\beta}|_0 < p^{\alpha} \) with difficulty.

Difficult to improve to \( |\hat{\beta}|_0 \leq p \) since then all the heavy lifting is being done by the accuracy claims.
What to do?

- Nothing will ever work perfectly
- So we have to hope the world is nice to us
- Let’s trust in this hope.
New algorithm: Alpha investing

Algorithm summary:
- Sort the variables putting the ones you like best first
  - For example, linear terms before interactions
  - put variables used last year before new ones to try
- Try each variable one at a time
- Add it to the regression if it is significant
  - Simplest rule: keep any with $|t| > \sqrt{2 \log(p)}$
  - Fancy rule: Use alpha spending. But, give yourself an $\alpha$ bonus ever time you reject.

Alpha investing algorithm

```plaintext
Wealth = .05;
while (Wealth > 0) do
    bid = amount to bid;
    Wealth = Wealth - bid;
    let X be the next variable to try;
    if (p-value of X is less than bid) then
        Wealth = Wealth + .05;
        Add X to the model
    end
end
```
New algorithm: Alpha investing

- This is even more Greedy than stepwise regression
- provides mFDR protection
- Instead of orthogonalizing each new $X$, only approximately orthogonalize it.
  - Can be done via sampling
  - Can be done use fast matrix methods
- For sub-modular problems, it works well
VIF speed comparison

Capacity

- vif-regression
- gps
- stepwise
- lasso
- foba

Number of Candidate Variables

Elapsed Running Time

Capacity

- vif:100,000
- gps:6,000
- stepwise:900
- lasso:700
- foba:600
Out−of−sample Error −− Comparison of Different Algorithms (p = 200)

θ = 0.1
θ = 0.3
θ = 0.5
θ = 0.7
θ = 0.9
Conclusions

- Stepwise regression when used correctly has good performance
  - include variables with $|t| > \sqrt{2 \log(p)}$
  - Use interactions
  - Use dummy’s for missing values
  - Use robust p-values

- Other fast alternatives
  - alpha investing (this talk)
  - Fast matrix methods (last week’s talk)
  - gradient methods (yichaolu@ or try VW)
Stepwise regression when used correctly has good performance
- include variables with $|t| > \sqrt{2 \log(p)}$
- Use interactions
- Use dummy’s for missing values
- Use robust p-values

Other fast alternatives
- alpha investing (this talk)
- Fast matrix methods (last week’s talk)
- gradient methods (yichaolu@ or try VW)

Thanks!
Streaming feature selection was introduced in JMLR 2006 (with Zhou, Stine and Ungar).
Let $W(j)$ be the “alpha wealth” at time $j$. Then for a series of p-values $p_j$, we can define:

$$W(j) - W(j - 1) = \begin{cases} \omega & \text{if } p_j \leq \alpha_j, \\ -\alpha_j/(1 - \alpha_j) & \text{if } p_j > \alpha_j. \end{cases}$$

(1)

**Theorem**

*(Foster and Stine, 2008, JRSS-B)* An alpha-investing rule governed by (1) with initial alpha-wealth $W(0) \leq \alpha \eta$ and pay-out $\omega \leq \alpha$ controls $mFDR_\eta$ at level $\alpha$. 
VIF regression

(Foster, Dongyu Lin, 2011) VIF regression approximates a streaming feature selection method with speed $O(np)$. 

Theorem
(Foster, Johnson, Stine, 2013) If the R-squared in a regression is submodular (aka subadditive) then a streaming feature selection algorithm will find an estimator whose out risk is within a factor of $e/(e - 1)$ of the optimal risk.
Wealth = .05;
while (Wealth > 0) do
    bid = amount to bid;
    Wealth = Wealth - bid;
    let X be the next variable to try;
    if (p-value of X is less than bid) then
        Wealth = Wealth + .05;
        Add X to the model
    end
end
mFDR for streaming feature selection

Streaming feature selection was introduced in JMLR 2006 (with Zhou, Stine and Ungar).
Let $W(j)$ be the “alpha wealth” at time $j$. Then for a series of p-values $p_j$, we can define:

$$W(j) - W(j-1) = \begin{cases} \omega & \text{if } p_j \leq \alpha_j, \\ -\alpha_j/(1-\alpha_j) & \text{if } p_j > \alpha_j. \end{cases}$$

(1)

VIF regression

Theorem

(Foster, Dongyu Lin, 2011) VIF regression approximates a streaming feature selection method with speed $O(np)$.

Submodular

Theorem

(Foster, Johnson, Stine, 2013) If the R-squared in a regression is submodular (aka subadditive) then a streaming feature selection algorithm will find an estimator whose out risk is within a factor of $e/(e-1)$ of the optimal risk.

Bibliography
**Bibliography**

**bibliography: risk inflation**


**bibliography: Streaming feature selection**

- Kory Johnson, Bob Stine, Dean Foster “Submodularity in statistics.”

**bibliography: Computational issues**

- Justin Thaler, Howard Karloff, and Dean Foster, “L-0 regression is hard.”
- Moritz Hardt, Jonathan Ullman “Preventing False Discovery in Interactive Data Analysis is Hard.”


Kory Johnson, Bob Stine, Dean Foster “Submodularity in statistics.”


Justin Thaler, Howard Karloff, and Dean Foster, “L-0 regression is hard.”

Moritz Hardt, Jonathan Ullman “Preventing False Discovery in Interactive Data Analysis is Hard.”
A success for stepwise regression

**Theorem (Natarajan 1995)**

*Stepwise regression will have a prediction accuracy of at most twice optimal using at most $\approx 18|X^+|_2^2 q$ variables.*

- This result was only recently noticed to be about stepwise regression. He didn’t use that term.
- The risk inflation is a disaster.
- The $|X^+|_2$ is a measure of co-linearity.
- This bound can be arbitrarily large.
- Brings up two points: we are willing to “cheat” on both accuracy and number of variables. But hopefully not by very much.
L0 regression is hard

Theorem (Zhang, Wainwright, Jordan 2014)
There exists an design matrix $X$ such that no polynomial time algorithm which outputs $q$ variables achieves a risk better than

$$R(\hat{\theta}) \gtrsim \frac{1}{\gamma^2(X)} \sigma^2 q \log(p).$$

Where $\gamma$ is the RE, a measure of co-linearity.

- Actual statement is much more complex and involves a version of the assumption that $P \neq NP$.
- It was previously known that that

$$R(\hat{\theta}_{lasso}) \lesssim \frac{1}{\gamma^2(X)} \sigma^2 q \log(p).$$
Theorem (Foster, Karloff, Thaler 2014)

No algorithm exists which achieves all three of the following goals:

- Runs efficiently (i.e. in polynomial time)
- Runs accurately (i.e. risk inflation < p)
- Returns sparse answer (i.e. $|\hat{\beta}|_0 \ll p$)

- Strongest version requires an assumption about complexity (which I can’t understand).
- The proof relies on “interactive proof theory.” (which I also can’t understand).
- The sparsity results depend on the assumptions used. We can get $|\hat{\beta}|_0 < cq$ easily, and $|\hat{\beta}|_0 < p^{.99}$ with difficulty.
- Difficult to improve to $|\hat{\beta}|_0 \leq p$ since then all the heavy lifting is being done by the accuracy claims.