Optimization of Discrete Choice Models

Gael Lederrey, Virginie Lurkin and Michel Bierlaire

Work in progress
Optimization of Discrete Choice Models?
DCMs Softwares

- Larch *(Newman et al., 2018)*
- MNLogit *(statsmodel, Python)*
- Biogeme *(Bierlaire, 2003)*
Motivation

Twitter users sent 456,000 tweets

Youtube users uploaded 300 hours of videos

Instagram users shared 46,740 pictures

Google received over 3,600,000 search queries

Every minute of the Day (2017)

Ref: Domo + Internet
What can we do?

1. Avoid using more data
   (Do we really need more data?)

2. Use more powerful computers
   (Do you know about Moore’s law?)

3. Stop using DCMs and use ML
   (Basically, it’s the same thing, no?)

4. Actually do something about DCMs
Where to get inspiration?

- Machine Learning is the obvious choice!
  - Emerging since 1950’s
  - Lot’s of work on Optimization thanks to Neural Networks
  - They make use of data (data-driven)
    => they know how to deal with data!

ML is actually “close” to DCMs
DCMs

Model-driven

Main goal: Understand behavior

Can also predict

Likelihood as objective function

Optimization is very important!

v.s.

ML

Data-driven

Main goal: Prediction

Can also help to understand behavior

Likelihood (possible) as objective function

Optimization is very important!
One fundamental difference

The data!
Optimization of ML - The Basics

**GD**  
(Cauchy, 1847)

<table>
<thead>
<tr>
<th>Specificities:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient computed on all the data</td>
</tr>
</tbody>
</table>

**Update step:**  
$$\theta = \theta - \alpha \cdot \nabla_\theta f(\theta; x)$$

<table>
<thead>
<tr>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\theta): Parameters</td>
</tr>
<tr>
<td>(\alpha): Step size</td>
</tr>
<tr>
<td>(f): Function, (f \in C^1(\mathbb{R}^n))</td>
</tr>
<tr>
<td>(x): Data, (x \in \mathbb{R}^n)</td>
</tr>
</tbody>
</table>

**SGD**  
(???, 1940’s)

<table>
<thead>
<tr>
<th>Specificities:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient computed on only one data</td>
</tr>
</tbody>
</table>

**Update step:**  
$$\theta = \theta - \alpha \cdot \nabla_\theta f(\theta; x_i)$$

<table>
<thead>
<tr>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\theta): Parameters</td>
</tr>
<tr>
<td>(\alpha): Step size</td>
</tr>
<tr>
<td>(f): Function, (f \in C^1(\mathbb{R}^n))</td>
</tr>
<tr>
<td>(x): Data, (x \in \mathbb{R}^n)</td>
</tr>
</tbody>
</table>

**mbSGD**  
(???, 1940’s)

<table>
<thead>
<tr>
<th>Specificities:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gradient computed on a batch of data</td>
</tr>
</tbody>
</table>

**Update step:**  
$$\theta = \theta - \alpha \cdot \nabla_\theta f(\theta; x_{\sigma(k)})$$

<table>
<thead>
<tr>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\theta): Parameters</td>
</tr>
<tr>
<td>(\alpha): Step size</td>
</tr>
<tr>
<td>(f): Function, (f \in C^1(\mathbb{R}^n))</td>
</tr>
<tr>
<td>(x): Data, (x \in \mathbb{R}^n)</td>
</tr>
<tr>
<td>(\sigma(k)): Choice of (k) indices</td>
</tr>
</tbody>
</table>

First-order methods
A lot of work!

Adagrad (Duchi et al., 2011)

RMSprop (Hinton, 2012)

Momentum (Polyak, 1964)

NAG (Nesterov, 1983)

Adadelta (Zeiler, 2012)

Adam (Kingma & Ba, 2015)

Nadam (Dozat, 2016)

AdaMax (Kingma & Ba, 2015)

AMSGrad (Reddi et al., 2018)

SAGA (Defazio et al., 2014)

SAG (Schmidt et al., 2013)

Averaging (Polyak & Juditsky, 1992)
First-Order vs Second-Order

- Gradient is pretty cheap to compute
  \[ \nabla_{\theta} f(\theta; x) \in \mathbb{R}^d \]

- Computation of Hessian is difficult/impossible
  \[ \nabla^2 f(\theta; x) \in \mathbb{R}^{d \times d} \]

- Recently, more work on quasi-Newton methods.
What am I doing?
Newton Method

- Update step:
  \[ x_{n+1} = x_n - \alpha [\nabla^2 f(x_n)]^{-1} \nabla f(x_n) \]

- Use Conjugate Gradient to find the step direction
  \[ \nabla^2 f(x_n) \Delta x = -\nabla f(x_n) \]

- Use Line Search for proper step size
  - Wolfe 1, Wolfe 2, Armijo, etc.
Trust Region

- Define a trust region around current point: $x_n$

- Use Taylor to approximate $f(x_n)$

$$f(x_n + \Delta x) = f(x_n) + \nabla f(x_n) \Delta x + \frac{1}{2} \nabla^2 f(x_n)(\Delta x)^2$$

- Find $\Delta x$ that minimize the Taylor approx.

- Based on $\Delta x$, decide to:
  - Augment the size of the TR
  - Reduce the size of the TR
  - Do a step or try again
  - ...
Models and datasets

- Swissmetro (~10k) and MNL
- Swissmetro (~10k) and Nested Logit Model
- Bike Sharing (~16k) and Logistic Regression
Newton Method and Trust Region

Bike Sharing

Nested

MNL
Introducing Stochasticity

- No “Full” gradient or Hessian
- Choose a “final” batch size
- Sample the data without replacement
Is Stochasticity a good thing?

Batch size: 1000

Bike Sharing

Nested

MNL
Adaptive Batch Size (ABS)
ABS at work!

Bike Sharing

Nested

MNL
Comparison

Bike Sharing

Nested

MNL
**Wrap-up**

**Newton Method**
- Basic
- Stocha
- ABS

**Trust Region**
- Basic
- Stocha
- ABS
Conclusion

● Basic stochastic algorithms seem to struggle

● A lot of work is needed on the batch size

● **But**, Promising early results!
Future work

● Testing ABS on
  ○ More models: Cross Nested, Mixed Logit, etc.
  ○ More data: RP & SP data from 2015 microsensus

● Continue developing ideas about
  ○ Different batch size update
  ○ Quasi Newton methods?

● Candidacy!
Thank you!