Optimization of DCMs using first-order methods

Optimization of DCMs

And beyond!

Gael Lederrey
PhD student @ TRANSP-OR, EPFL

05.09.2018

Outline

1. Motivation
2. State-of-the-art
3. Optimization of DCMs
4. Beyond Optimization
5. Conclusion

Motivation

DCMs in a nutshell

- Well established theory with many success stories!

Why is a solution required?

BIG DATA

Every minute of the Day (2017)

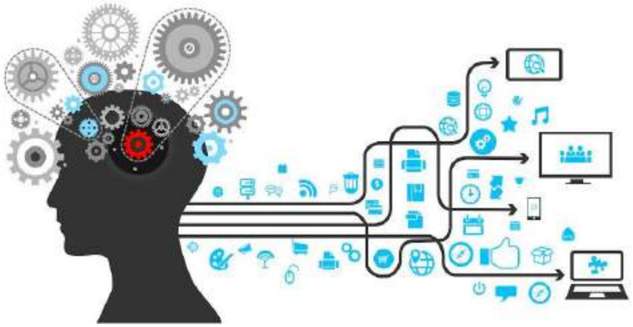
Twitter users sent
456,000 tweets

Youtube users uploaded
400 hours of video

Instagram users shared **46,740 pictures**

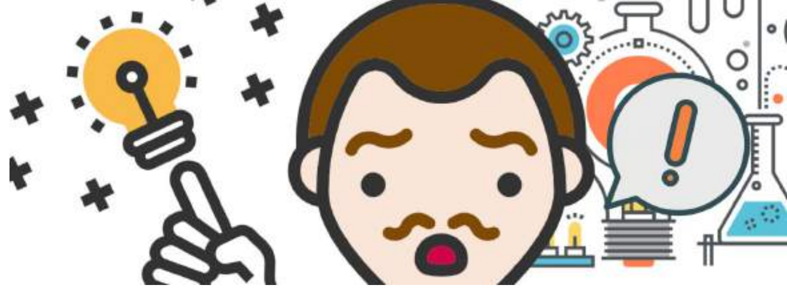
Google received over
3,600,000 search queries

Solution: Machine Learning



- ML is **THE** field dealing with a lot of data!

What can we do for DCMs?



1. Faster optimization of DCMs.

State-of-the-art

First-order optimization - the ancestors

GD

(Cauchy, 1847)

```
Specificities:
  Gradient computed on
  all the data
```

Update step:

$$\theta = \theta - \alpha \cdot \nabla_\theta f(\theta; x)$$

Where

θ : Parameters

α : Step size

f : Function, $f \in C^1(\mathbb{R}^n)$

x : Data, $x \in \mathbb{R}^n$

SGD

(???, 1940's)

```
Specificities:
  Gradient computed on
  only one data
```

Update step:

$$\theta = \theta - \alpha \cdot \nabla_\theta f(\theta; x_i)$$

Where

θ : Parameters

α : Step size

f : Function, $f \in C^1(\mathbb{R}^n)$

x : Data, $x \in \mathbb{R}^n$

mbSGD

(???, 1940's)

```
Specificities:
  Gradient computed on
  a batch of data
```

Update step:

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

Where

θ : Parameters

α : Step size

f : Function, $f \in C^1(\mathbb{R}^n)$

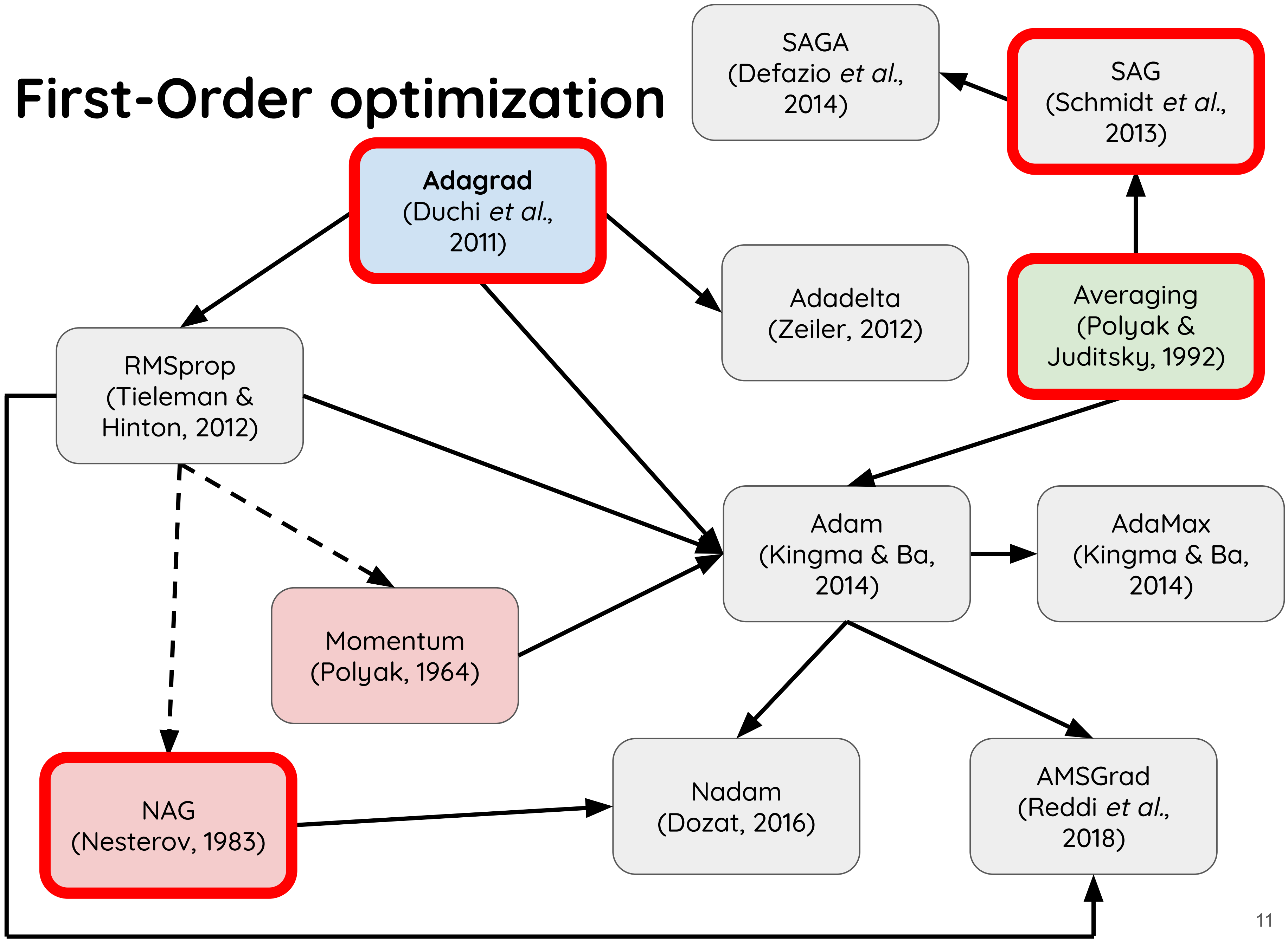
x : Data, $x \in \mathbb{R}^n$

$\sigma(k)$: Choice of k indices

Challenges

- Choosing a proper step size
- Same step size applies to all parameter updates
- **Avoid getting trapped in a local minima!**
  (For non-convex functions)

First-Order optimization



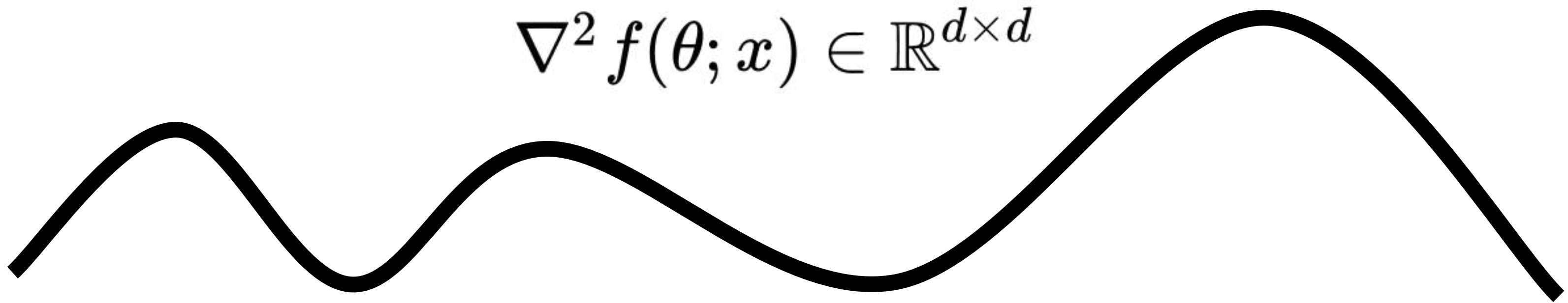
First-Order vs Second-Order

- 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.

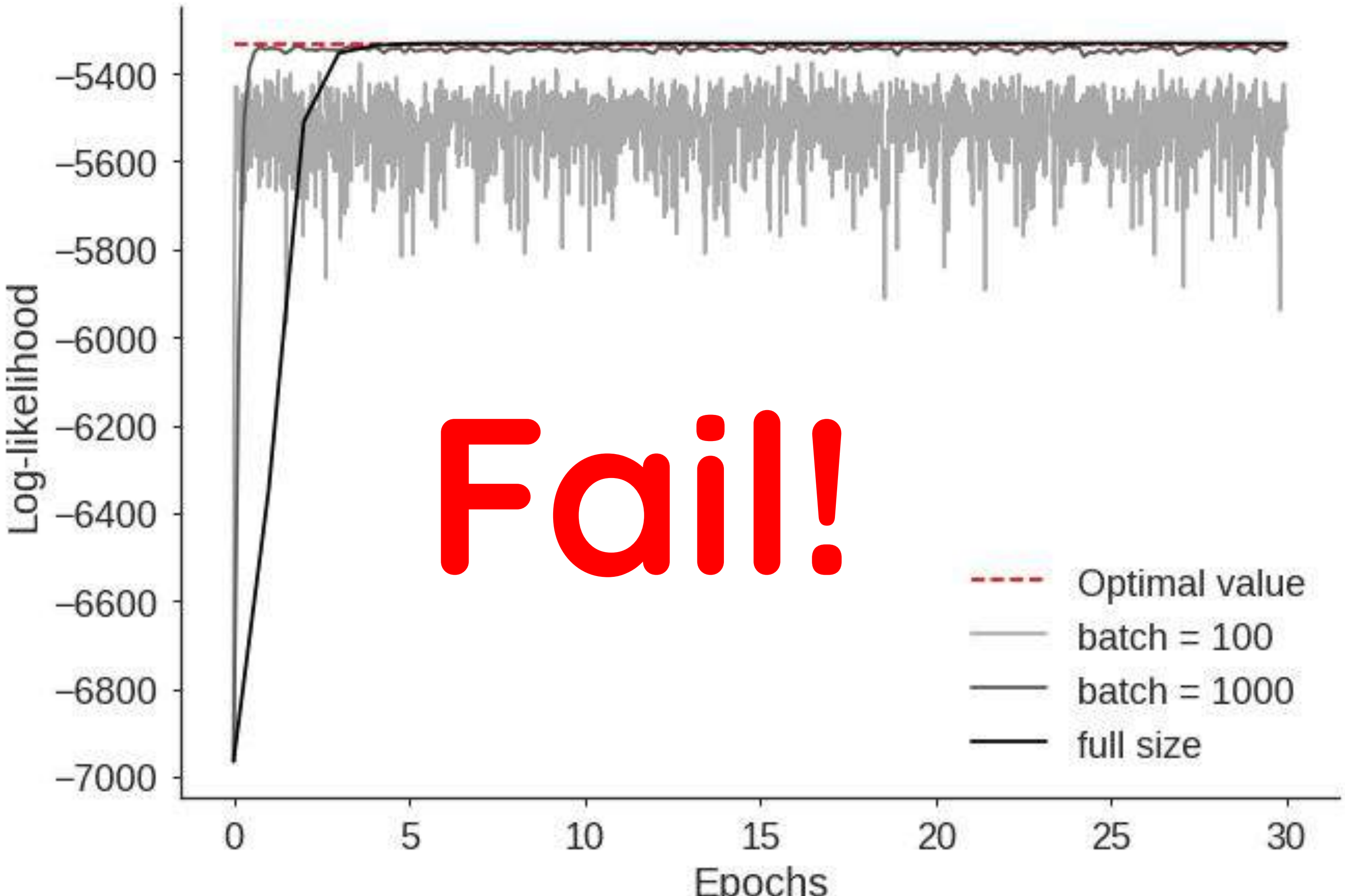
  Gower *et al.* (2018), Martens (2010), Bordes *et al.* (2009-2010)

Optimization of Discrete Choice Models

Motivation

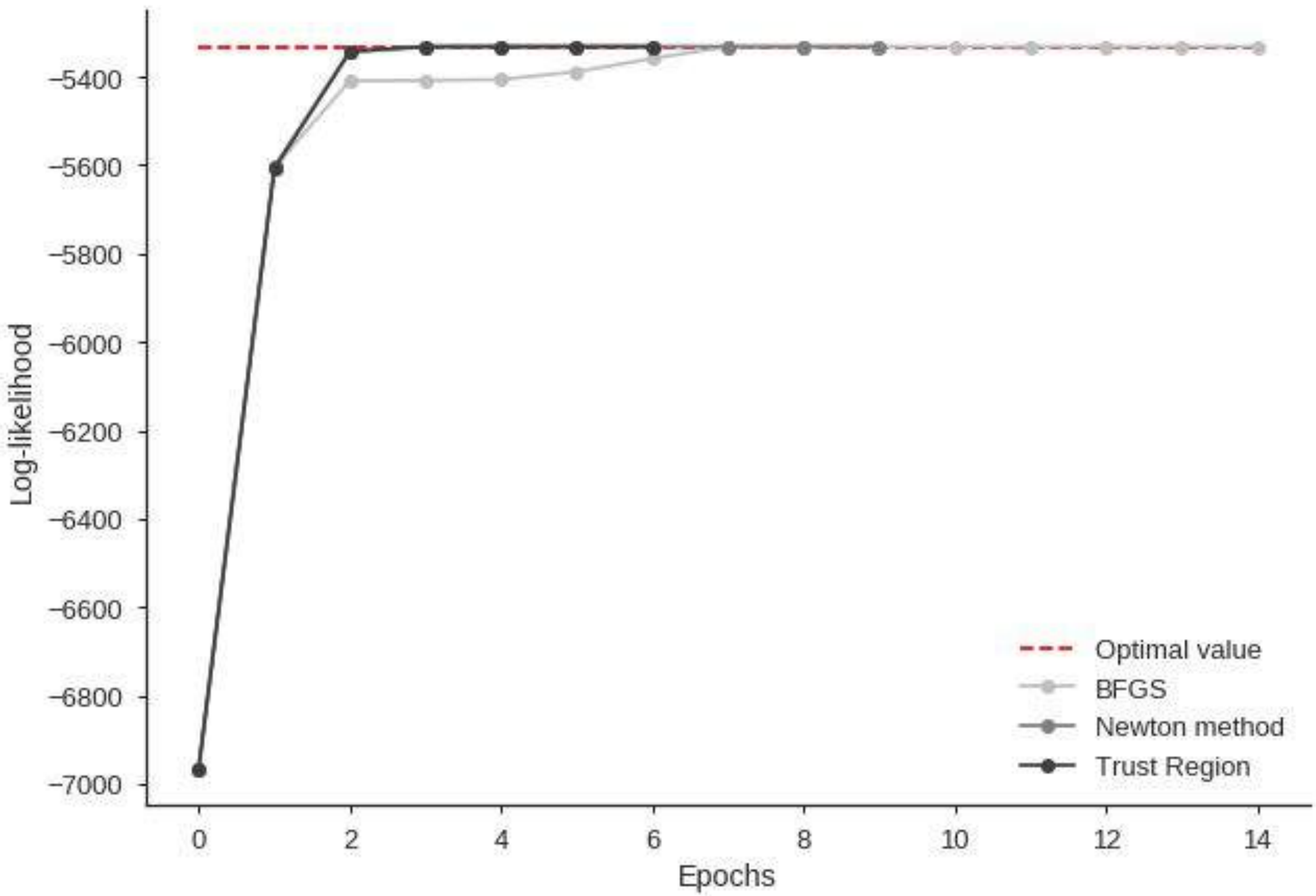
- Data is growing everyday!
- DCMs (will) have to deal with these new datasets!
- => Make them faster, especially with big datasets.

First-order methods

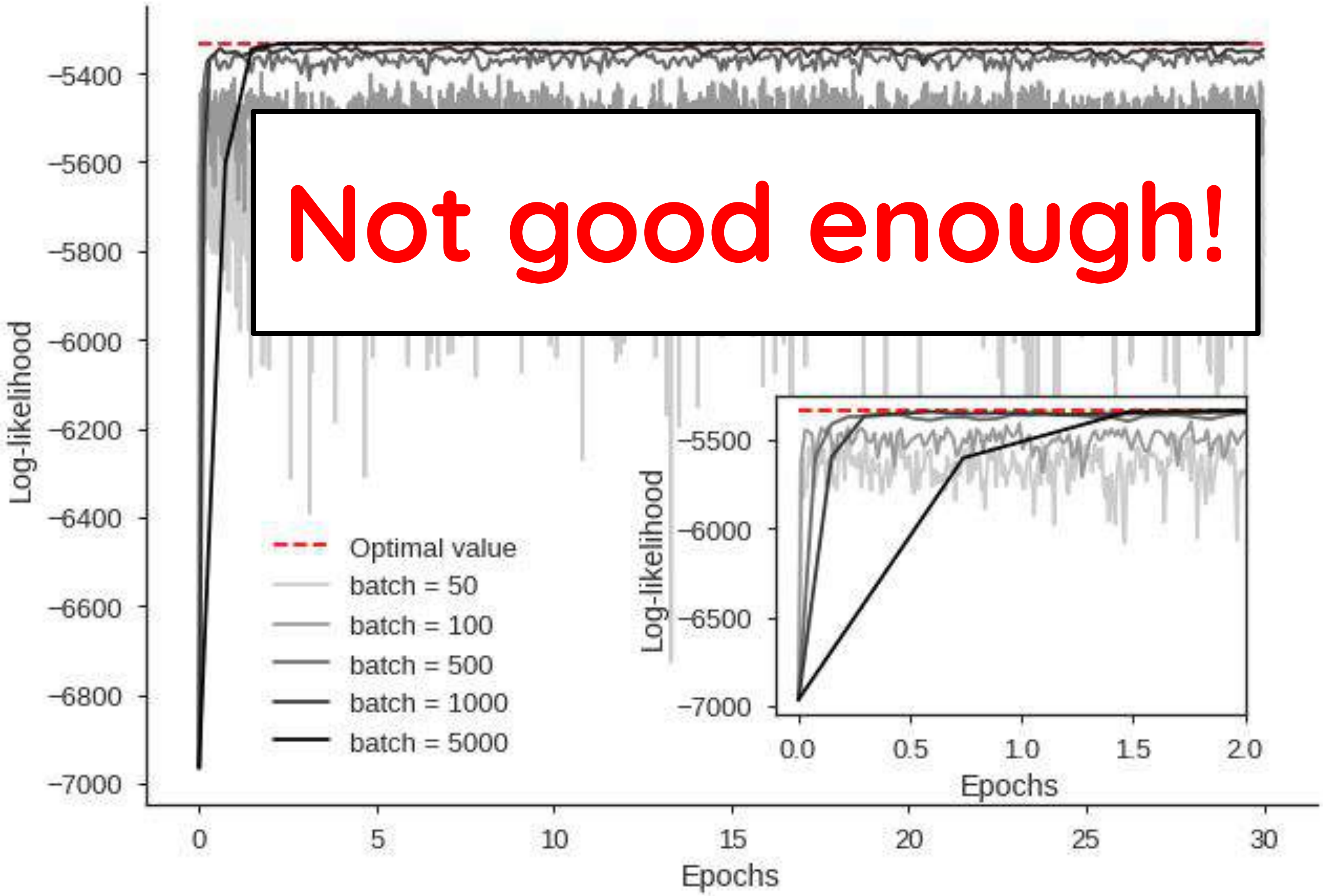


Lederrey *et al.*, 2018 - hEART 2018

Second-order methods



Stochastic Newton Method



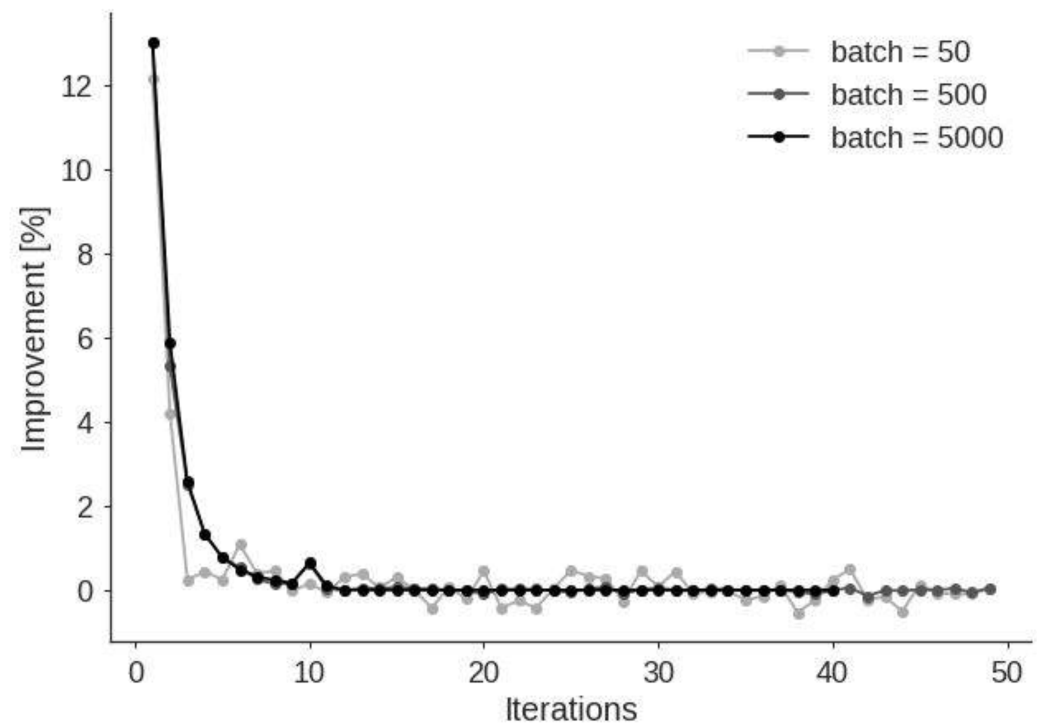
Not good enough!

t

Lederrey *et al.*, 2018 - IEEE ITSC 2018

Problems?

- Seems to be stuck around the optimum.
  => "Flat area"?



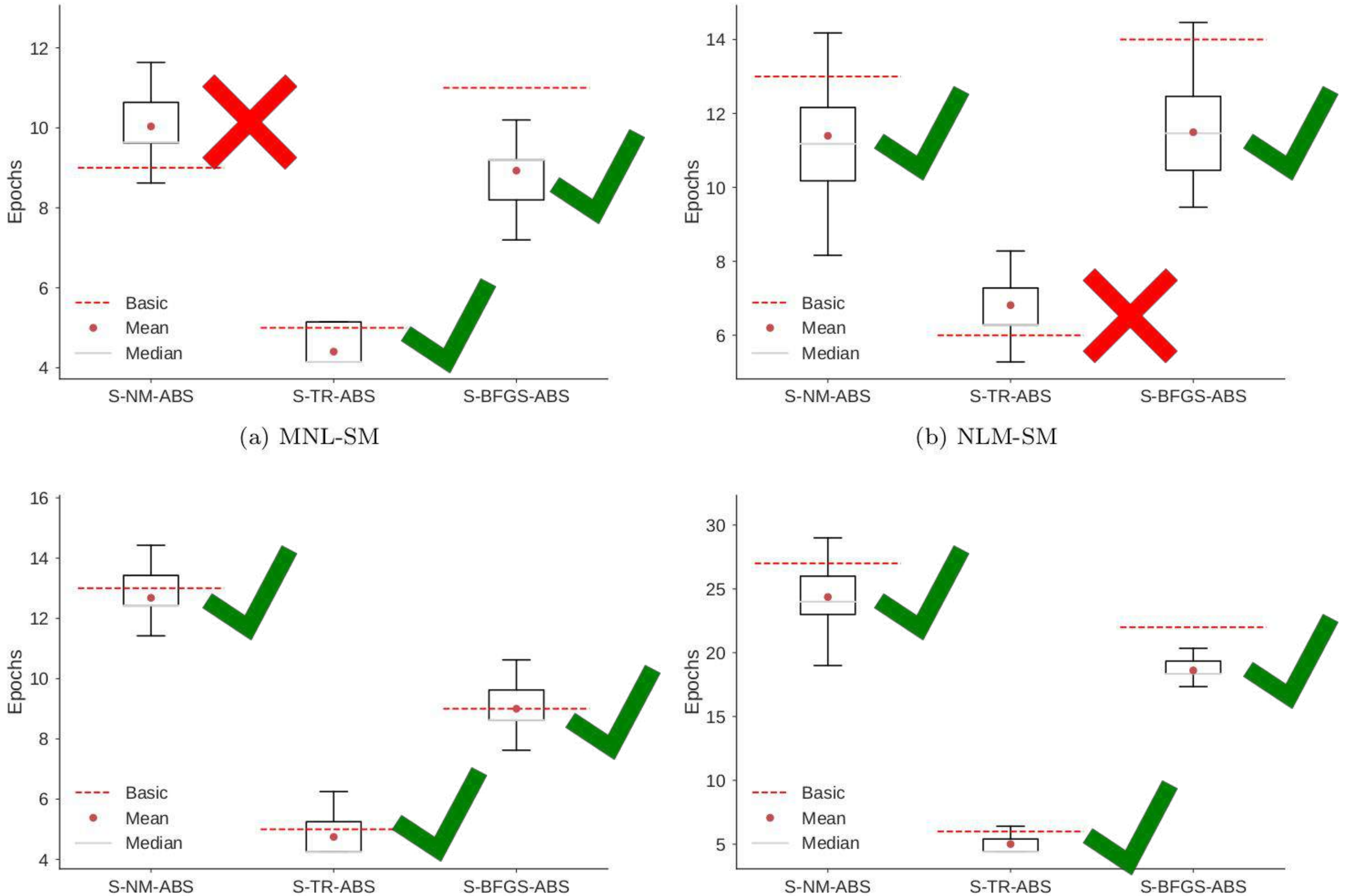
Adaptive Batch Size (ABS)

coming soon!

- Not enough improvement => Update the BS!

Results

Better: 10/12



(a) MNL-SM

(b) NLM-SM

(c) LogReg-BS

(d) LogReg-BS-Full

Next steps

- Test the ABS technique on bigger models/datasets
  => Danalet and Mathys (2018)

Suggestions are welcome!

- **Final step:** Write an article!

Beyond Optimization

What's left?

1. ~~Faster optimization of DCMs.~~
2. Better at predicting. (⚠️ **Interpretability**)
3. Use data to find "better" models. (faster!)

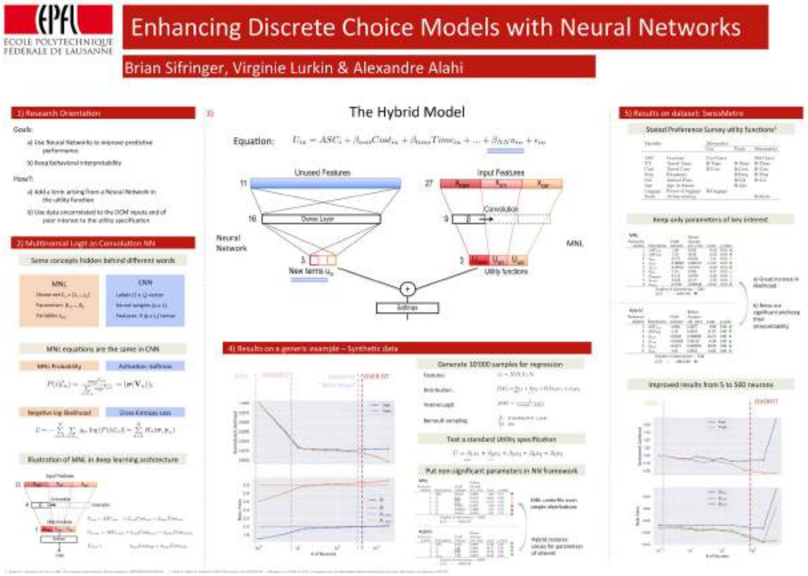
} ML

DCMs	v.s.	ML
Model-driven		Data-driven
Main goal: Understand behavior		Main goal: Prediction
Can also predict		Can also help to understand behavior
Likelihood as objective function		Likelihood (possible) as objective function
Optimization is very important!		Optimization is very important!

What we should not do!

What should we do then?

Friday, 11:00-12:00



- Improve DCMs with ML!
- Examples:

Conclusion

Conclusion

- DCMs have to be improved!
- Better optimization process are already working!
- ML can bring a lot to the DCMs field...
- The same goes in the other direction.

How can we help each other?

Do you have trouble optimizing huge and complicated DCMs with big datasets?

Contact me:

gael.lederrey@epfl.ch

Thank you!

Back up
Slides

ABS Algorithm

Algorithm 1 Adaptive Batch Size (ABS)

Input: Current iteration index (M), function value at iteration M (f_M), and batch size (n)
Output: New batch size (n')

function ABS
    Store f_M in a list $\mathcal{F}$
    Compute $WMA_{M,W}$ using $\mathcal{F}$ and store it in a list $\mathcal{A}$
    if $M > 0$ **then** ▷ We need at least two values to compute the improvement.
        Compute i the improvement as in Equation 3 using the list $\mathcal{A}$ and store it in a list $\mathcal{I}$
        if $n < N$ **then**
            if $\mathcal{I}_M < \Delta$ **then** ▷ Improvement under the threshold
                $c = c + 1$
            else
                $c = 0$ ▷ We restart the counter
            if $c == C$ **then** ▷ We will update the batch size
                $c = 0$ ▷ We restart the counter
                $n' = \tau \cdot n$
                if $n' >= N$ **then** ▷ The batch size is too big now
                    $n' = N$
            else
                $n' = n$
    return n'