Hyper-parameters Selection by Automatic Differentiation

Samuel Vaiter

Joint works with

Patrice Abry, Quentin Bertrand, Mathieu Blondel, Jérome Bolte, Charles Deledalle, Charles Dossal, Jalal Fadili, Alexandre Gramfort, Quentin Klopfenstein, Mathurin Massias, Barbara Pascal, Edouard Pauwels, Gabriel Peyré, Nelly Pustelnik, Joseph Salmon

Parametric estimators

Estimator

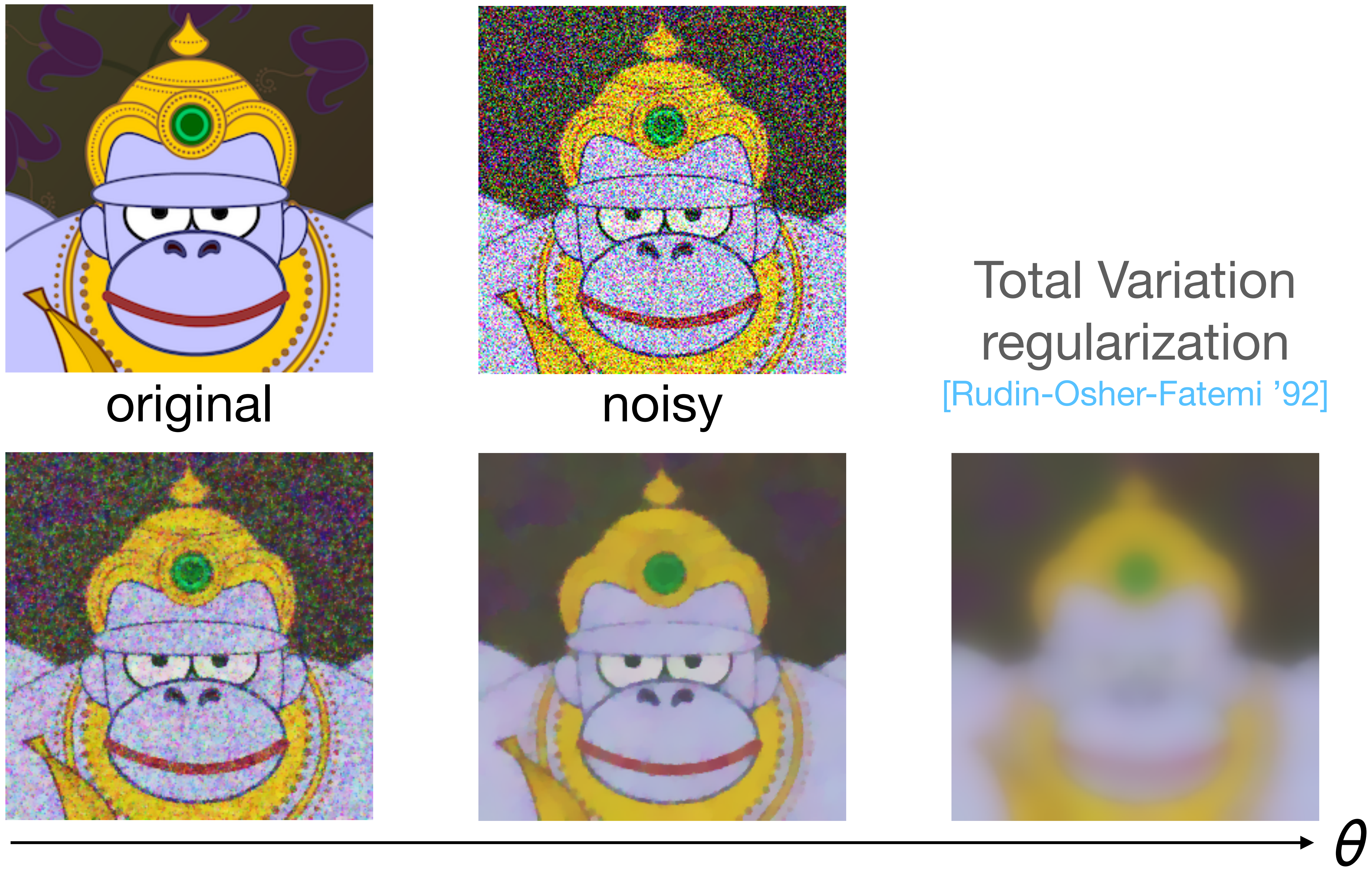
$$\hat{w}_\theta : \begin{cases} \mathbb{R}^n & \longrightarrow \mathbb{R}^p \\ y & \longmapsto \hat{w}_\theta(y) \end{cases}$$

observations

Hyper-parameters

$$\theta \in \Theta \subseteq \mathbb{R}^l$$

parameter space



Typical example

linear regression

$$y = X w_{\text{true}} + \varepsilon$$

design matrix, ground truth, noise

regularization *a.k.a* variational methods

$$\hat{w}_\theta(y) \in \operatorname{argmin}_w \text{datafit}(w, y) + \text{regularity}(w, \theta)$$

trade-off

e.g. Lasso: $\operatorname{argmin}_w \|y - Xw\|_2^2 + \theta \|w\|_1$ [Chen-Donoho '94, Tibshirani '95]

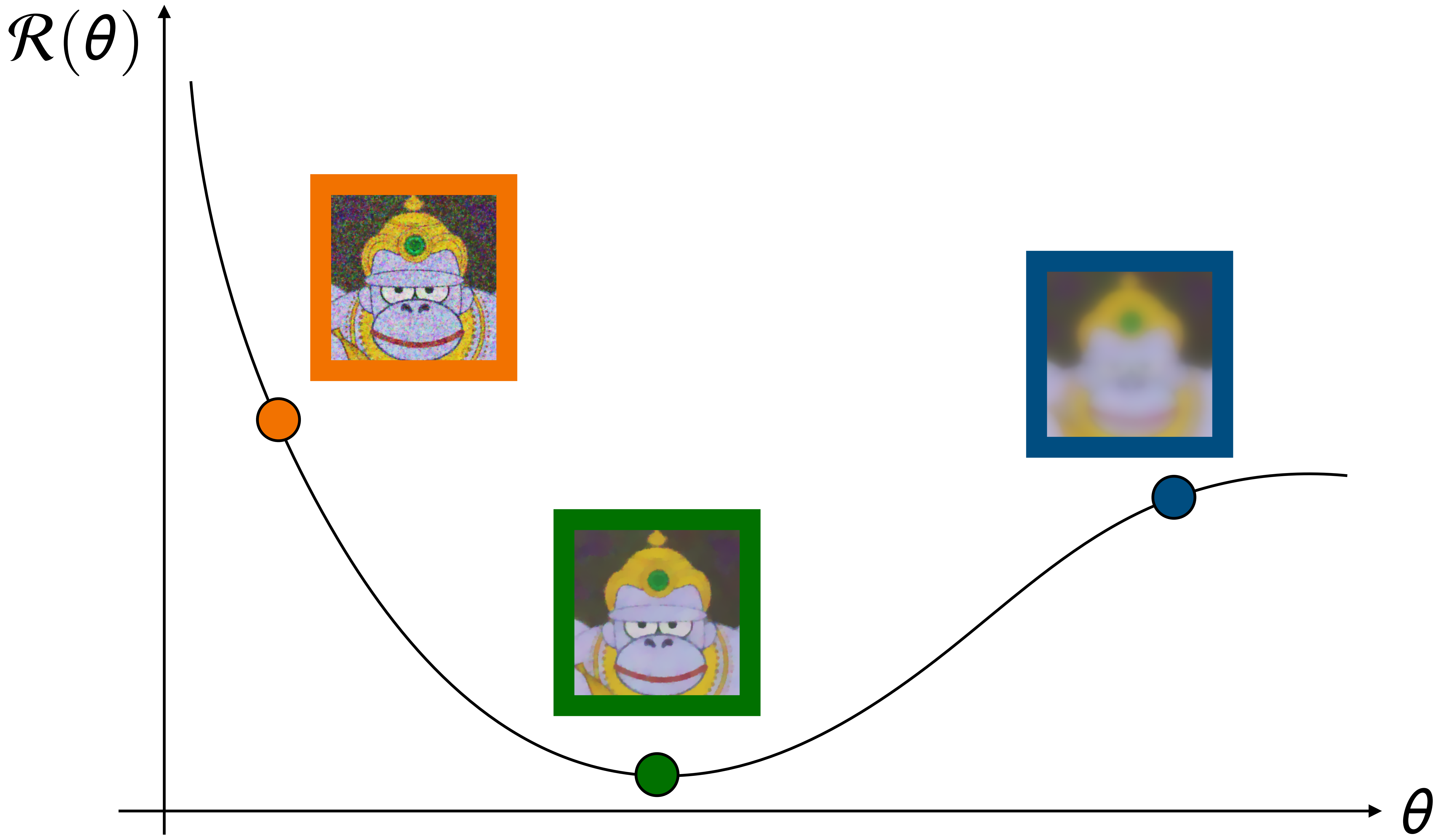
Selection criteria

Estimator

$\hat{w}_\theta : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ estimator

$\theta \in \Theta$ hyper-parameter

$\mathcal{R} : \Theta \longrightarrow \mathbb{R}$ criterion



Goal

Find $\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}}\ \mathcal{R}(\theta)$

(or close to it)

Inverse problems

$y = X w_{\text{true}} + \varepsilon$

estimation risk

[Rice '86]

$$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| \hat{w}_\theta(y) - w_{\text{true}} \|_2^2 \right)$$

Machine learning

validation set: $y^{\text{val}}, X^{\text{val}}$

hold-out loss

[Stone-Ramer '65]

$$\mathcal{R}(\theta) = \| y^{\text{val}} - X^{\text{val}} \hat{w}_\theta(y) \|_2^2$$

projected risk
cross-validation
model criteria
...

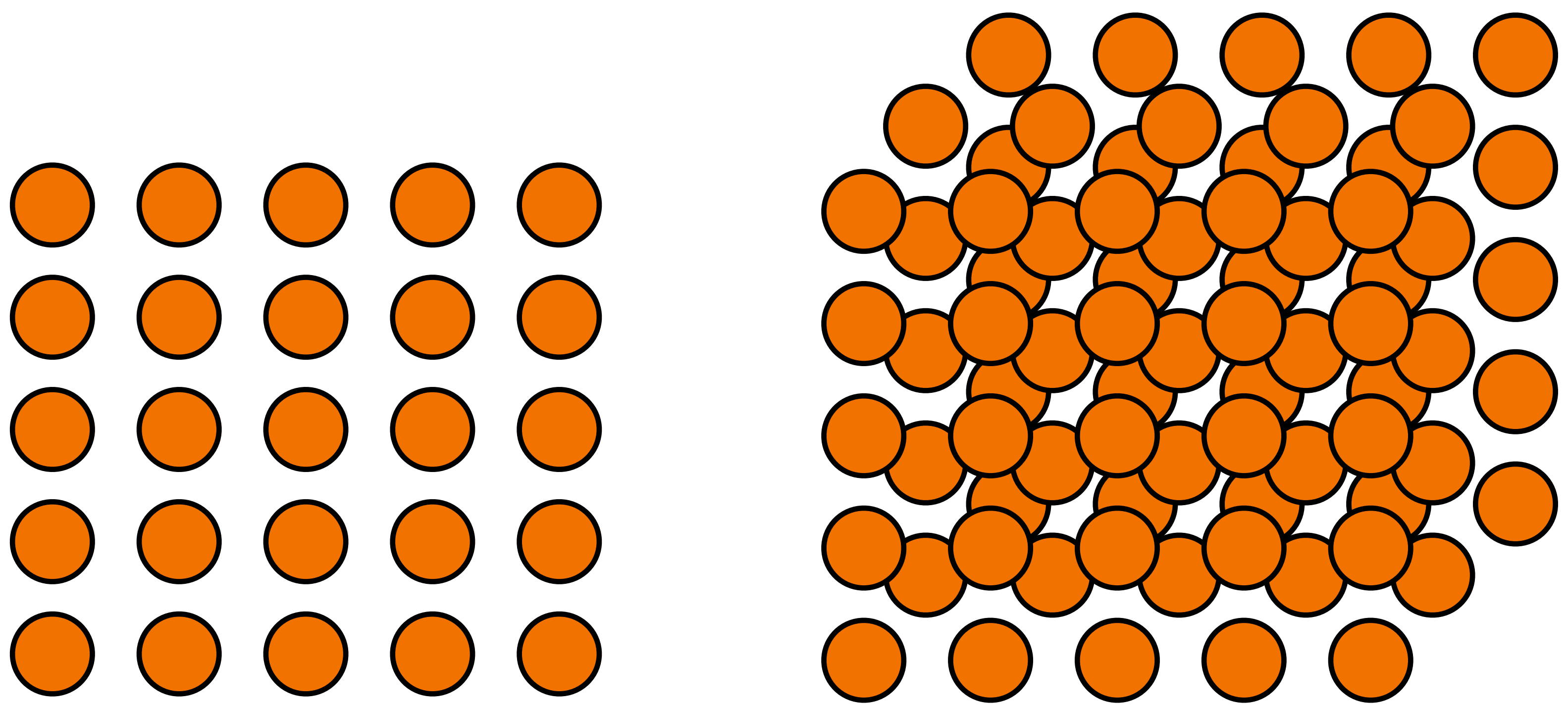
Grid search

Algorithm

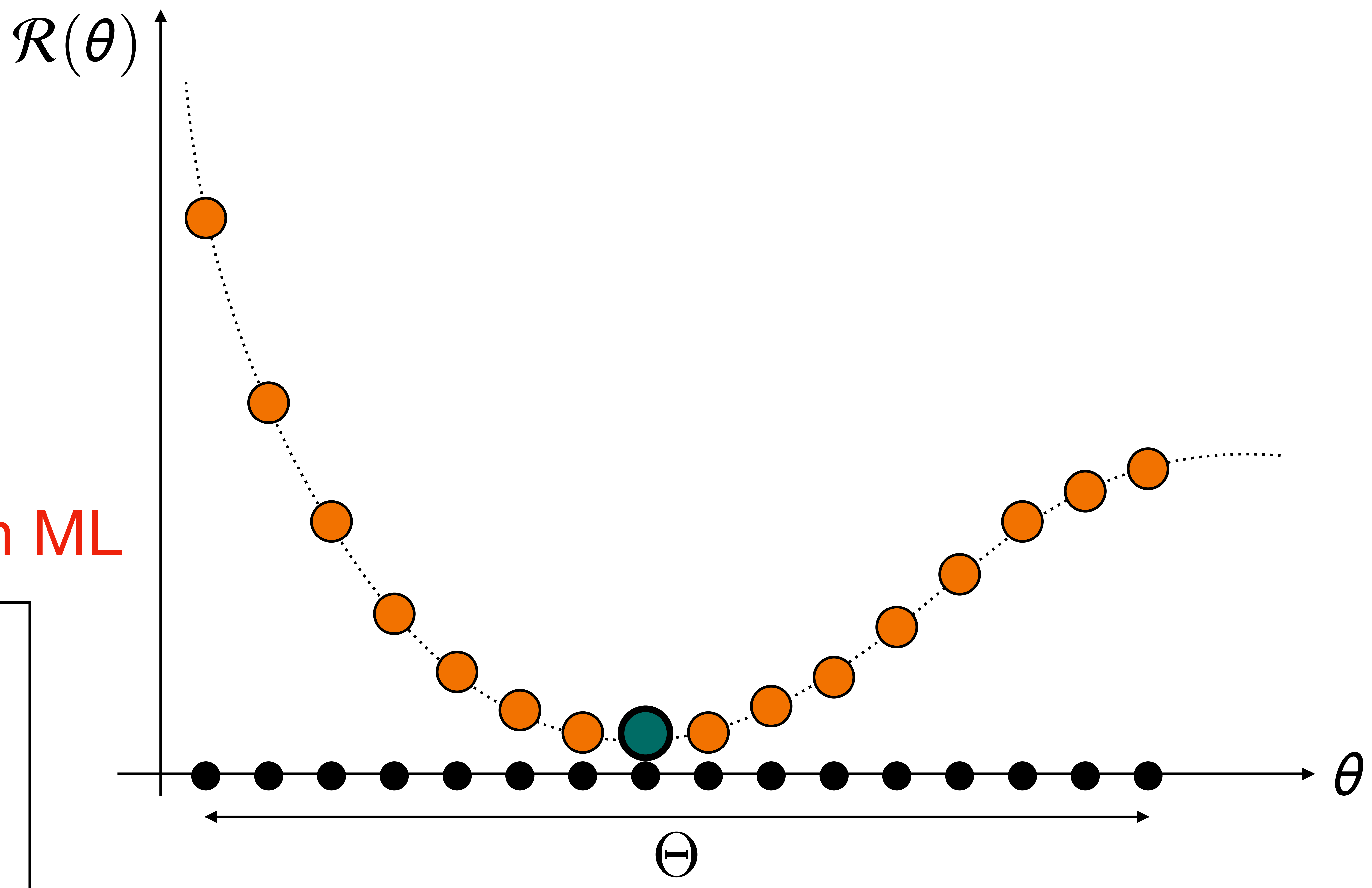
1. Choose a criterion $\mathcal{R}$
2.
3.
4.

Grid search is the standard in ML

Dimensionality issues of sampling Θ



2 parameters   3 parameters



Can be mitigated using Random Search [Bergstra-Bengio '12]
Bayesian methods [Brochu et al. '10]

First order methods for parameter selection

Goal

Find $\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}}\, \mathcal{R}(\theta)$

(or close to it)

$$\mathcal{R}(\theta) = \mathcal{C}(\hat{w}_\theta(y)) = (\mathcal{C} \circ \hat{w}_\bullet(y))\,(\theta)$$

$$\mathcal{C} : \mathbb{R}^p \to \mathbb{R}$$

$$\hat{w}_\bullet(y) : \Theta \subseteq \mathbb{R}^l \to \mathbb{R}^p$$

Chain rule

Assuming $\mathcal{C}$ and $\hat{w}_\bullet(y)$ to be differentiable

$$\nabla\mathcal{R}(\theta) = \left[\mathrm{Jac}_{\hat{w}_\bullet(y)}(\theta)\right]^\perp \nabla\mathcal{C}(\hat{w}_\theta(y))$$

and

$$\nabla\mathcal{R}(\theta^\star) = 0$$

Potential issues

differentiability
access to the cost
size of the Jacobian
approximation stability
convergence (non-convex)

“Hyper”-gradient descent

$$\theta^{k+1} = \theta^k - \rho\nabla\mathcal{R}(\theta^k)$$

possibly **projected** gradient descent on the parameter space

First order methods for parameter selection

Goal

Find $\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}}\, \mathcal{R}(\theta)$

(or close to it)

$$\mathcal{R}(\theta) = \mathcal{C}(\hat{w}_\theta(y)) = (\mathcal{C} \circ \hat{w}_{\bullet}(y))(\theta)$$

$$\mathcal{C} : \mathbb{R}^p \to \mathbb{R}$$

$$\hat{w}_{\bullet}(y) : \Theta \subseteq \mathbb{R}^l \to \mathbb{R}^p$$

Chain rule

Assuming $\mathcal{C}$ and $\hat{w}_{\bullet}(y)$ to be differentiable

$$\nabla \mathcal{R}(\theta) = \left[\mathrm{Jac}_{\hat{w}_{\bullet}(y)}(\theta)\right]^\perp \nabla \mathcal{C}(\hat{w}_\theta(y))$$

and

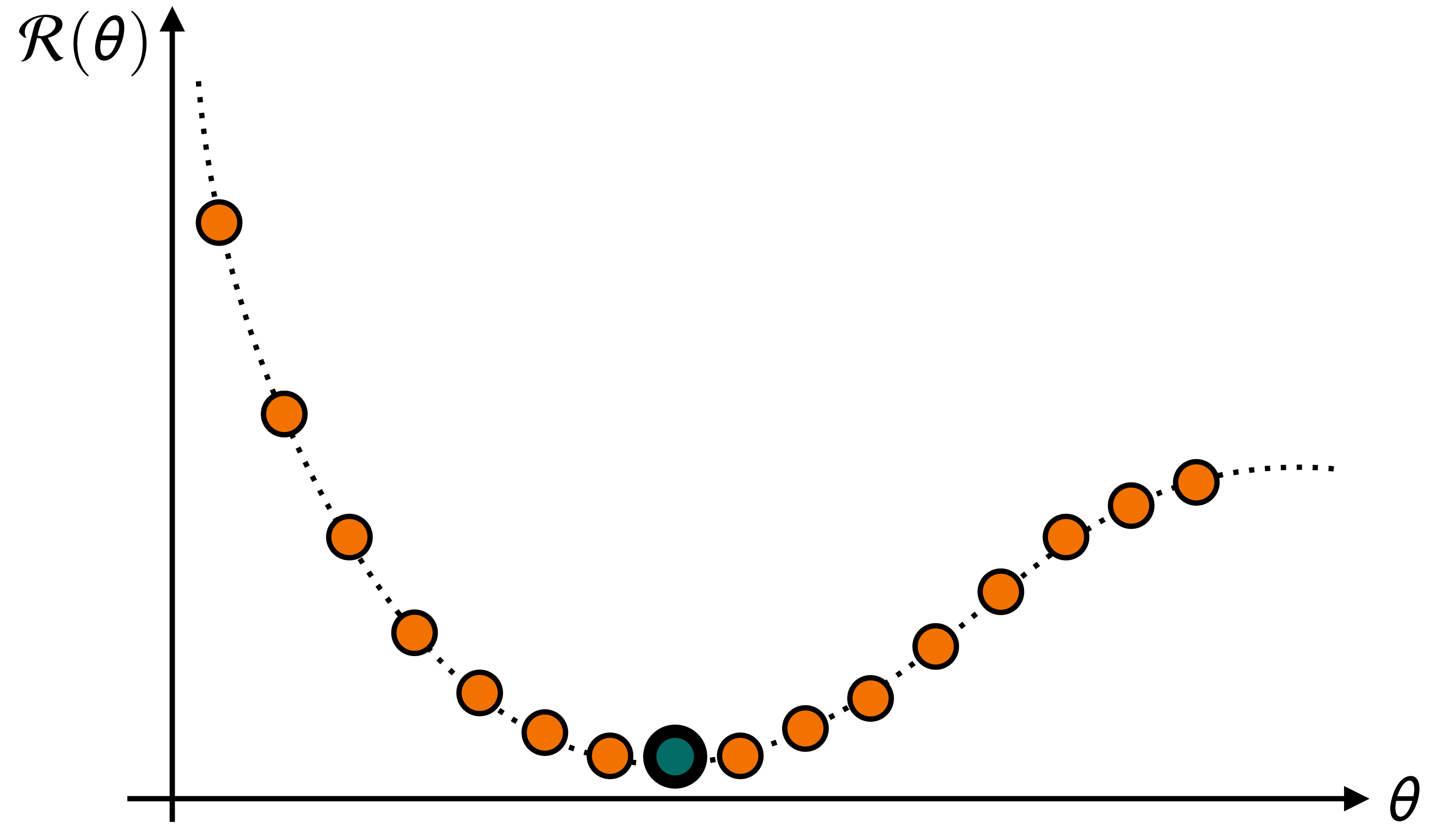
$$\nabla \mathcal{R}(\theta^\star) = 0$$

"Hyper"-gradient descent

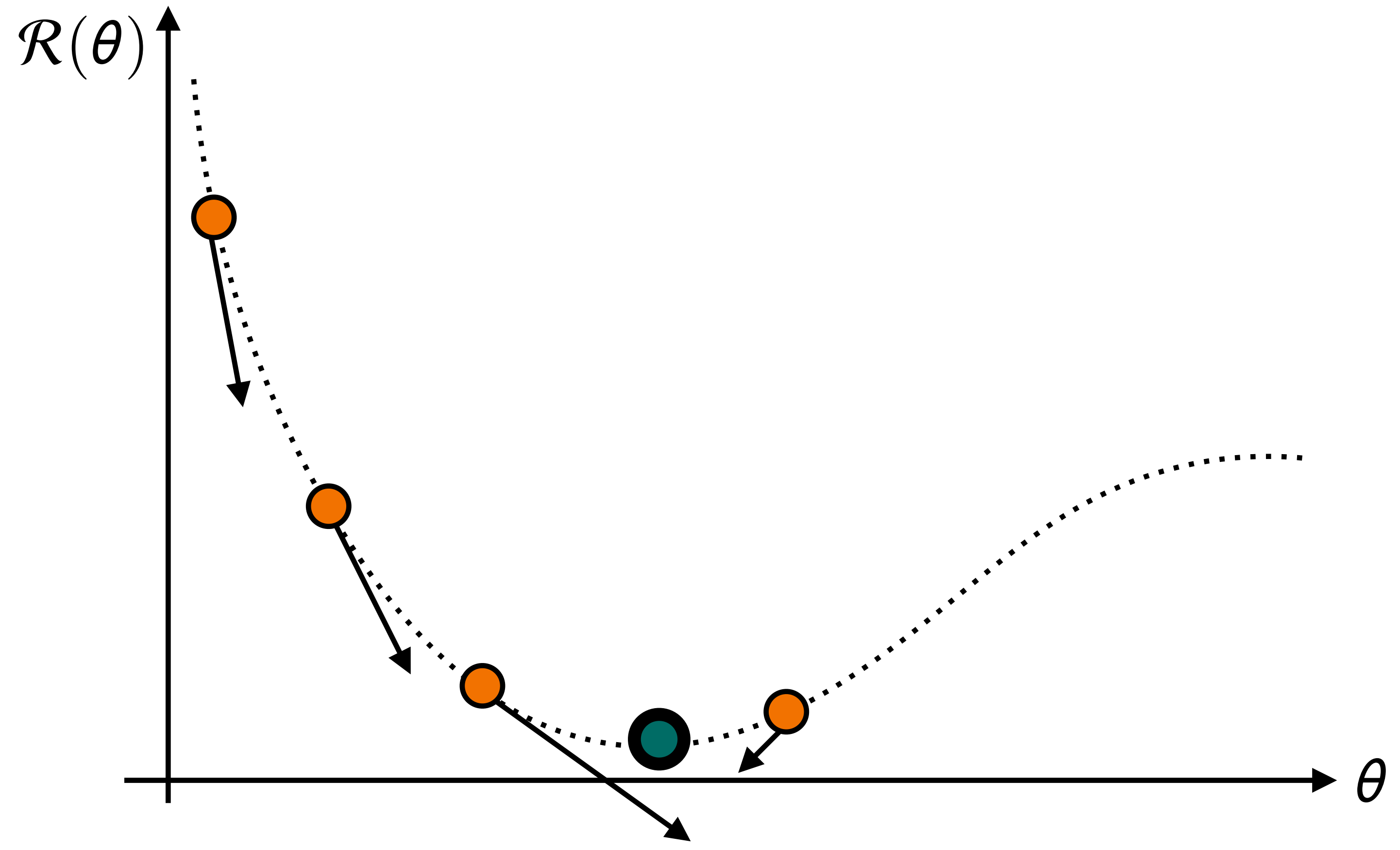
$$\theta^{k+1} = \theta^k - \rho \nabla \mathcal{R}(\theta^k)$$

possibly **projected** gradient descent on the parameter space

Grid search (a.k.a 0-order)



Gradient descent



Implicit differentiation

Q. Bertrand, Q. Klopfenstein, M. Blondel, SV, A. Gramfort, J. Salmon. Implicit differentiation of Lasso-type models for hyperparameter optimization. *ICML*. 2020.
Q. Bertrand, Q. Klopfenstein, M. Blondel, SV, A. Gramfort, J. Salmon. Implicit differentiation for fast hyperparameter selection in non-smooth convex learning. *JMLR*. 2022.

Bilevel problem: implicit differentiation

Goal

Find $\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}} \mathcal{R}(\theta)$

(or close to it)

$$\mathcal{R}(\theta) = \mathcal{C}(\hat{w}_\theta(y))$$

- $\mathcal{C}$, $\nabla \mathcal{C}$ easy to compute
- $\hat{w}_\theta$ variational estimator
- $\mathcal{F}$ convex smooth

Bilevel optimization

outer problem

$$\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}} \mathcal{C}(\hat{w}_\theta(y))$$

subject to $\quad \hat{w}_\theta(y) \in \underset{w \in \mathbb{R}^p}{\operatorname{argmin}} \mathcal{F}(w, \theta)$

inner problem

$$\nabla \mathcal{R}(\theta) = \left[\mathrm{Jac}_{\hat{w}_\bullet(y)}(\theta)\right]^\perp \nabla \mathcal{C}(\hat{w}_\theta(y))$$

[Larsen et al. '96] **Fixed point equation** $\quad \nabla_w \mathcal{F}(\hat{w}_\theta(y), \theta) = 0$

$\dfrac{\partial}{\partial \theta}$

$$\nabla^2_{w,\theta} \mathcal{F}(\hat{w}_\theta(y), \theta) + \left[\mathrm{Jac}_{\hat{w}_\bullet(y)}(\theta)\right]^\perp \nabla^2_w \mathcal{F}(\hat{w}_\theta(y), \theta) = 0$$

assuming invertibility

$$\left[\mathrm{Jac}_{\hat{w}_\bullet(y)}(\theta)\right]^\perp = -\nabla^2_{w,\theta} \mathcal{F}(\hat{w}_\theta(y), \theta) \left(\nabla^2_w \mathcal{F}(\hat{w}_\theta(y), \theta)\right)^{-1}$$

Smooth case

Kernel-based
[Chapelle et al. '02]
Weighted ridge
[Foo et al. '08]
Image restoration
[Kunish-Pock '13]
Noisy stability
[Pedregosa '16]

Non-smooth

Elastic-net
[Mairal et al. '12]
Lasso
[Dossal et al. '13,
Zou et al. '07]
Generalized Lasso
[V. et al. '13
Tibshirani-Taylor '11]
Partly smooth
[V. et al. '17]
Constrained quad
[Amos-Kolter '17]
Simplex constrained
[Niculae-Blondel '17]

Fast performance for Lasso-like methods

Machine learning

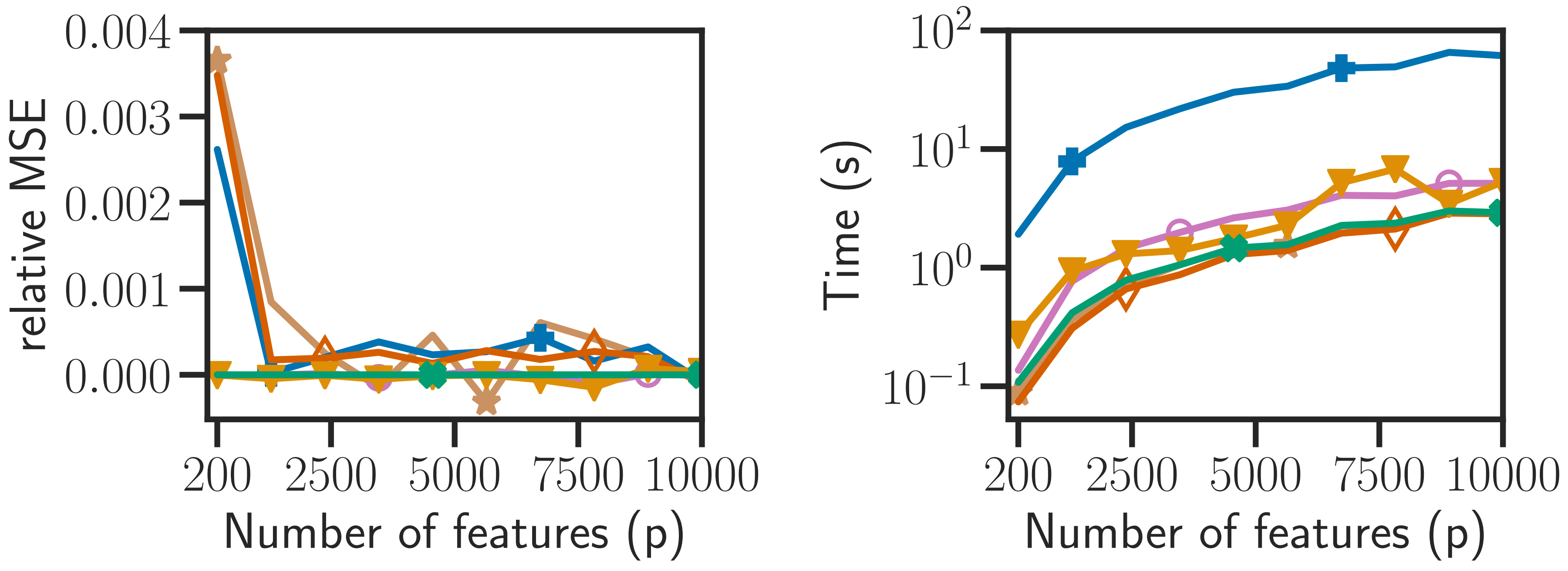
validation set: $y^{\text{val}}, X^{\text{val}}$

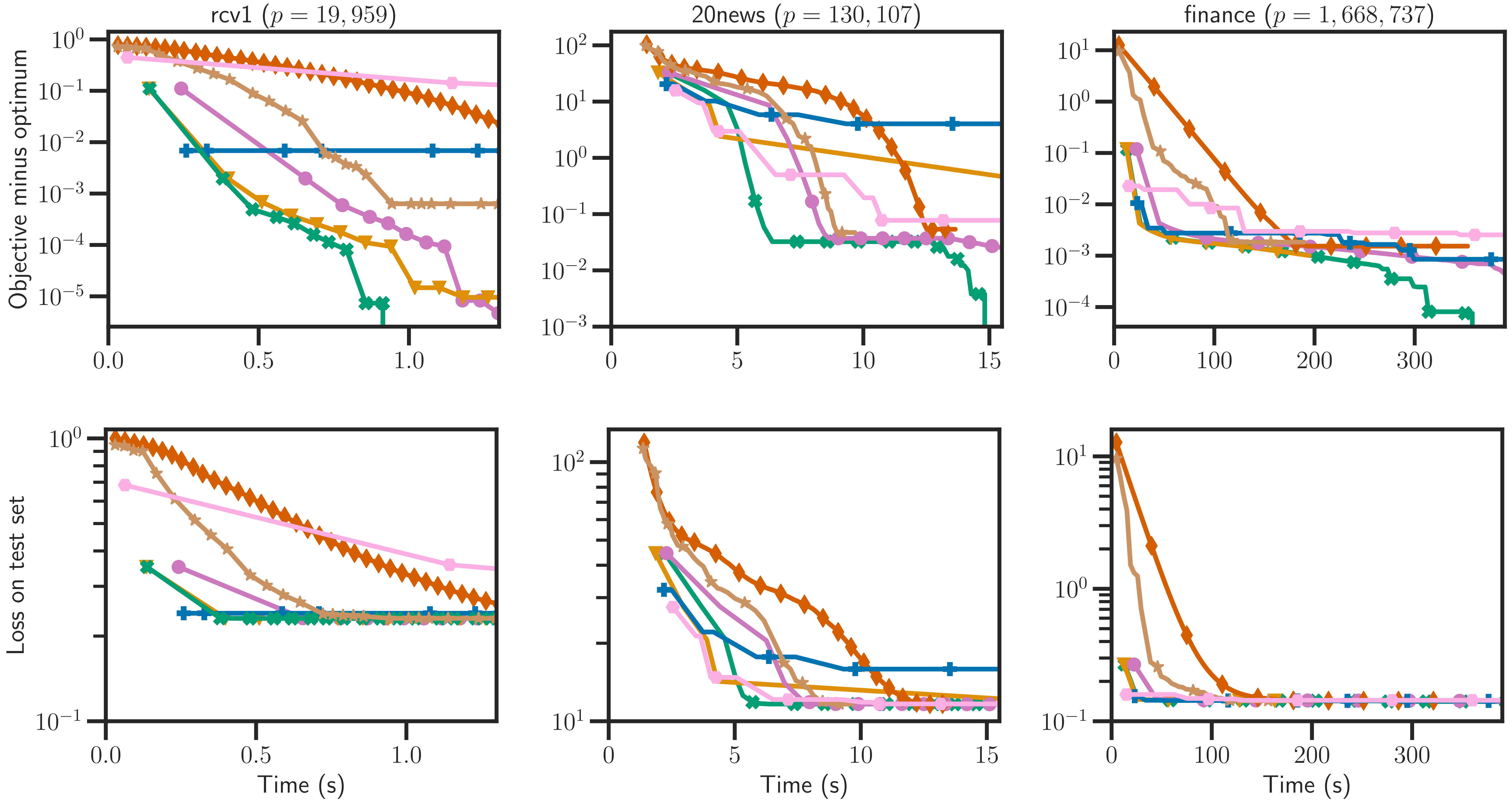
hold-out loss

[Stone-Ramer '65]

$$\mathcal{R}(\theta) = \|y^{\text{val}} - X^{\text{val}} \hat{w}_\theta(y)\|_2^2$$

$\hat{w}_\theta(y)$: Lasso





Computation time

Estimation performance

Estimating a Jacobian

Estimator

$\hat{w}_\theta : \mathbb{R}^n \longrightarrow \mathbb{R}^p$

differentiable

Jacobian

$\text{Jac}_{\hat{w}_\bullet}(y) : \Theta \longrightarrow \mathbb{R}^{p \times l}$

$\text{Jac}_{\hat{w}_\theta} : \mathbb{R}^n \longrightarrow \mathbb{R}^{p \times n}$

$$\text{Jac}_{\hat{w}_\theta}(y) = \begin{pmatrix} \frac{\partial(\hat{w}_\theta)_1}{\partial y_1}(y) & \cdots & \frac{\partial(\hat{w}_\theta)_1}{\partial y_n}(y) \\ \vdots & & \vdots \\ \frac{\partial(\hat{w}_\theta)_p}{\partial y_1}(y) & \cdots & \frac{\partial(\hat{w}_\theta)_p}{\partial y_n}(y) \end{pmatrix} \in \mathbb{R}^{p \times n}$$

Numerical approximation

$$\text{Jac}_{\hat{w}_\theta}(y) \approx \left(\frac{\hat{w}_\theta(y+\delta \mathbf{e}_1) - \hat{w}_\theta(y)}{\delta} \quad \cdots \quad \frac{\hat{w}_\theta(y+\delta \mathbf{e}_n) - \hat{w}_\theta(y)}{\delta} \right)$$

needs $O(nT)$ operations

parameter dimension — cost of evaluating $\hat{w}_\theta(\cdot)$

Symbolic differentiation

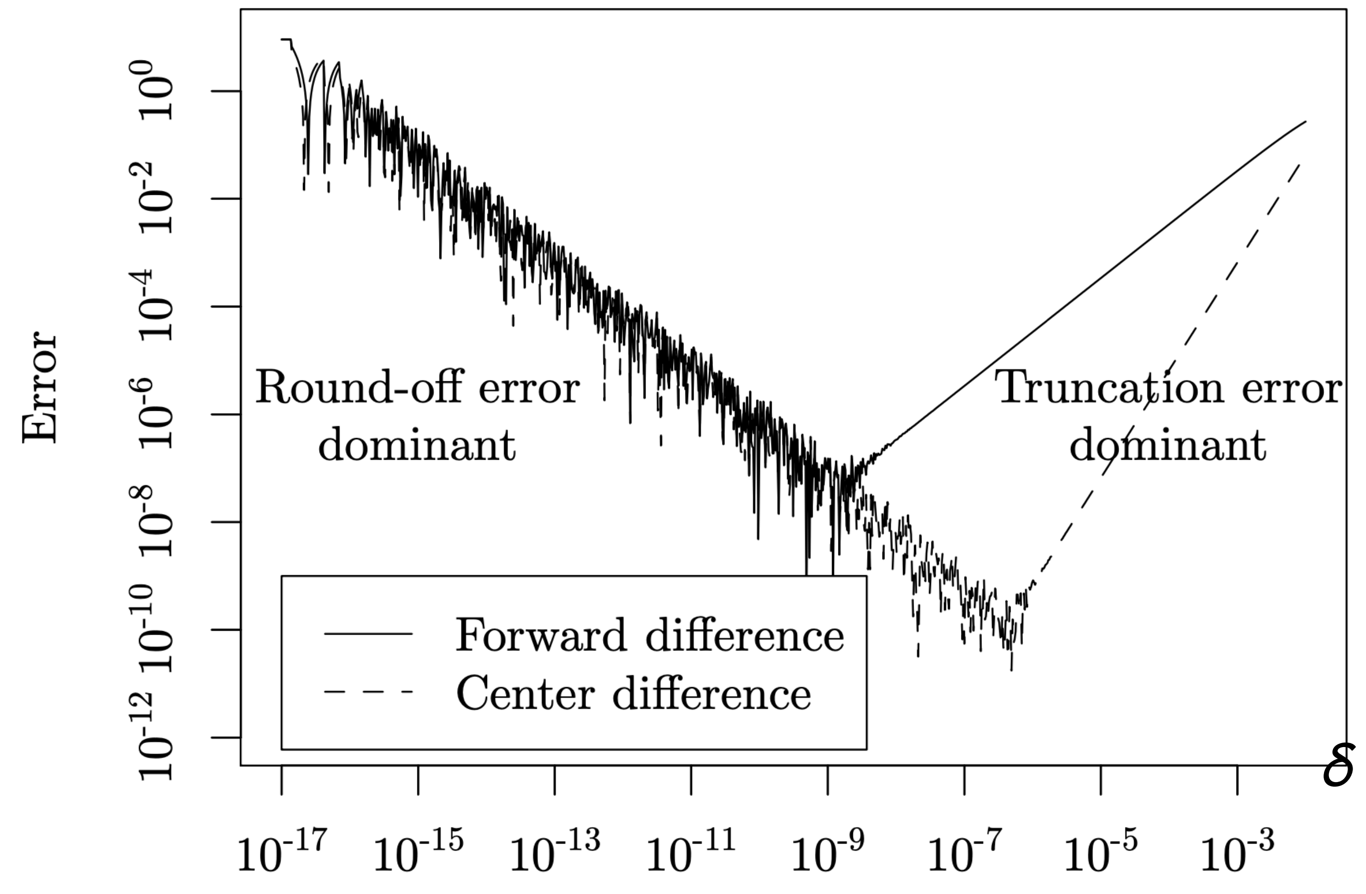
Used by Mathematica, Maple, SymPy, Maxima, Lisp

Ineffective for “complex” programs

$\longrightarrow$ expression swell

Numerical errors

from [Baydin ’18]



$f(x) = 64x(1-x)(1-2x)^2(1-8x+8x^2)^2$

Unstable for low **and** high precision

The SURE way

SV, C. Deledalle, G. Peyré, J. Fadili, C. Dossal. The Degrees of Freedom of Partly Smooth Regularizers. Ann Inst Stat Math. 69(4):791—832. 2017
SV, C. Deledalle, G. Peyré, C. Dossal, J. Fadili. Local Behavior of Sparse Analysis Regularization: Applications to Risk Estimation. Appl Comput Harmon Anal. 35(3):433—451. 2013.

Estimation and prediction risk

Goal

Find $\theta^\star \in \underset{\theta \in \Theta}{\operatorname{argmin}}\, \mathcal{R}(\theta)$

(or close to it)

Inverse problems

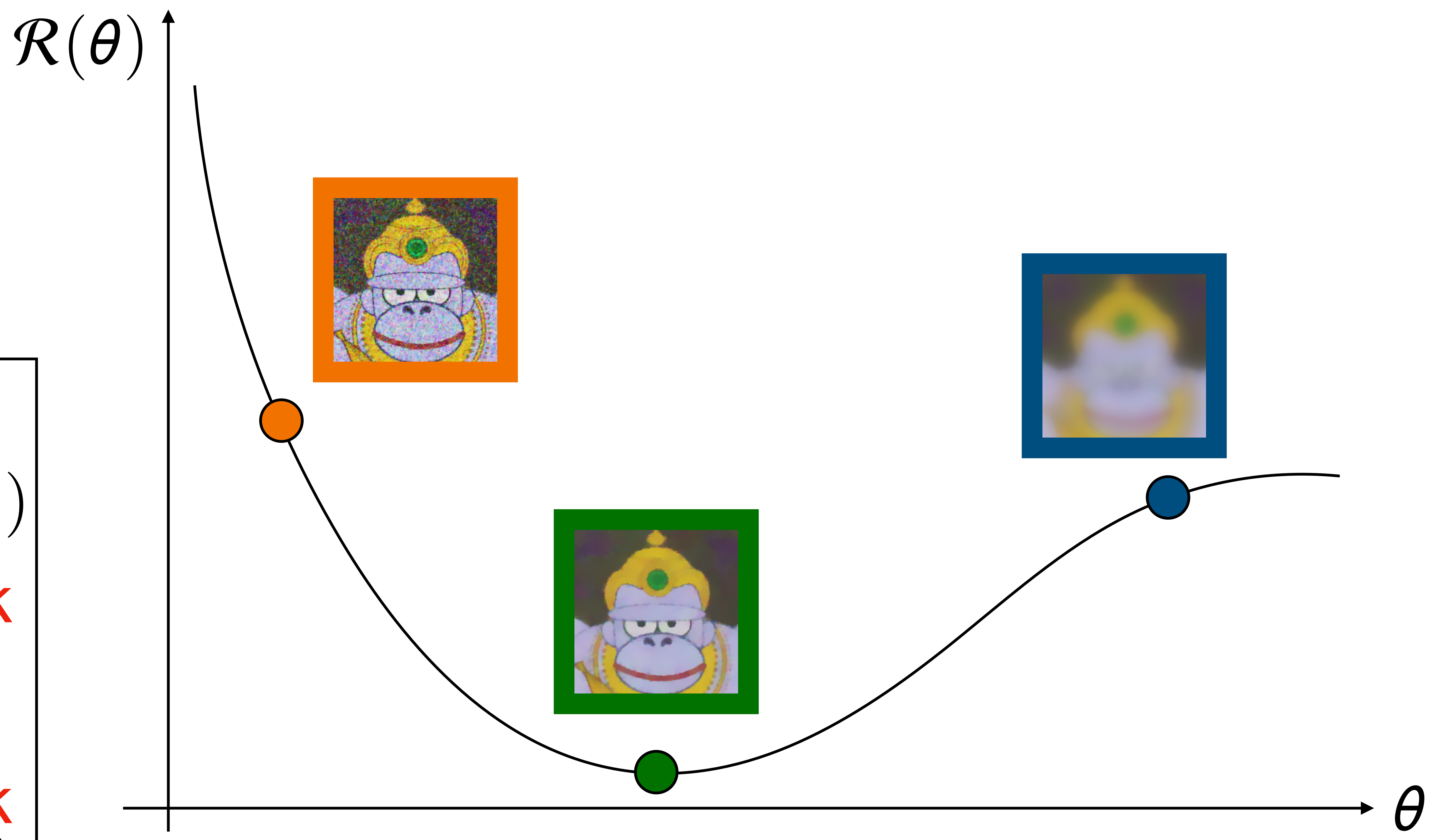
$y = X w_{\text{true}} + \varepsilon \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$

estimation risk

$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| \hat{w}_\theta(y) - w_{\text{true}} \|_2^2 \right)$

prediction risk

$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| X \hat{w}_\theta(y) - X w_{\text{true}} \|_2^2 \right)$



only **one** observation $y \implies \mathbb{E}_\varepsilon$ is not computable

only one **observation** $y \implies w_{\text{true}}$ is not known

Stein Unbiased Risk Estimation — SURE

Inverse problems

$$y = X w_{\text{true}} + \varepsilon \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$$

Prediction

$$\hat{\mu}_\theta(y) = X \hat{w}_\theta(y)$$

Prediction risk

$$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| \hat{\mu}_\theta(y) - X w_{\text{true}} \|_2^2 \right)$$

Degrees of freedom [Efron '86]

$$\text{df}_\theta(y) = \sum_{i=1}^{n} \frac{\text{cov}(\hat{\mu}_\theta(y)_i, y_i)}{\sigma^2}$$

→

Cp [Mallows '73]
AIC [Akaike '73]
BIC [Schwarz '78]
GCV [Craven-Wahba '79]
SURE [Stein '81]

Examples

Ordinary least square

$$\text{df}_\theta(y) = p$$

Lasso [Dossal et al. '13, Zou et al. '07]

$$\text{df}_\theta(y) = \| \hat{w}_\theta(y) \|_0 = |\text{supp}(\hat{w}_\theta(y))|$$

Stein's lemma

[Stein '81]

If $\hat{\mu}_\theta$ weakly differentiable

Empirical degrees of freedom

$$\widehat{\text{df}}_\theta(y) = \text{div}(\hat{\mu}_\theta(y)) = \sum_{i=1}^{n} \frac{\partial (\hat{\mu}_\theta)_i}{\partial y_i}(y)$$

$$\mathbb{E}_\varepsilon(\widehat{\text{df}}_\theta(y)) = \text{df}_\theta(y)$$

Stein Unbiased Risk Estimation

[Stein '81]

If $\hat{\mu}_\theta$ weakly differentiable

$$\text{SURE}_\theta(y) = \| y - \hat{\mu}_\theta(y) \|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\text{df}}_\theta(y)$$

$$\mathbb{E}_\varepsilon(\text{SURE}_\theta(y)) = \mathcal{R}(\theta)$$

🥴 requires the noise variance

SURE for smooth regularized least square

Inverse problems

$$y = X w_{\text{true}} + \varepsilon \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$$

Prediction

$$\hat{\mu}_\theta(y) = X \hat{w}_\theta(y)$$

Prediction risk

$$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\|\hat{\mu}_\theta(y) - X w_{\text{true}}\|_2^2 \right)$$

Degrees of freedom and SURE

$$\widehat{\mathrm{df}}_\theta(y) = \mathrm{div}(\hat{\mu}_\theta(y)) = \sum_{i=1}^{n} \frac{\partial (\hat{\mu}_\theta)_i}{\partial y_i}(y)$$

$$\mathrm{SURE}_\theta(y) = \|y - \hat{\mu}_\theta(y)\|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\mathrm{df}}_\theta(y)$$

$$\mathbb{E}_\varepsilon(\mathrm{SURE}_\theta(y)) = \mathcal{R}(\theta)$$

Smooth regularised least-square

$$\hat{w}_\theta(y) \in \underset{w \in \mathbb{R}^p}{\mathrm{argmin}} \frac{1}{2}\|y - X w\|_2^2 + \theta J(w)$$

First order conditions

$$X^\top (X \hat{w}_\theta(y) - y) + \theta \nabla J(\hat{w}_\theta(y)) = 0$$

Implicit function theorem

$$\Gamma_\theta(y) = X^\top X + \theta \nabla^2 J(\hat{w}_\theta(y)) \longrightarrow \mathrm{Jac}_{\hat{\mu}_\theta}(y) = X \Gamma_\theta(y)^{-1} X^\top$$

SURE for regularized least square

Inverse problems

$$y = X w_{\text{true}} + \varepsilon \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$$

Prediction

$$\hat{\mu}_\theta(y) = X \hat{w}_\theta(y)$$

Prediction risk

$$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\|\hat{\mu}_\theta(y) - X w_{\text{true}}\|_2^2 \right)$$

Degrees of freedom and SURE

$$\widehat{\text{df}}_\theta(y) = \text{div}(\hat{\mu}_\theta(y)) = \sum_{i=1}^{n} \frac{\partial (\hat{\mu}_\theta)_i}{\partial y_i}(y)$$

$$\text{SURE}_\theta(y) = \|y - \hat{\mu}_\theta(y)\|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\text{df}}_\theta(y)$$

$$\mathbb{E}_\varepsilon(\text{SURE}_\theta(y)) = \mathcal{R}(\theta)$$

~~Smooth~~ regularised least-square

$$\hat{w}_\theta(y) \in \underset{w \in \mathbb{R}^p}{\text{argmin}} \frac{1}{2}\|y - Xw\|_2^2 + \theta J(w)$$

Theorem [V. et al. '13,17]

When J is regular enough, $\text{Jac}_{\hat{\mu}_\theta(y)}(y)$ is computable *a.e.* in closed form

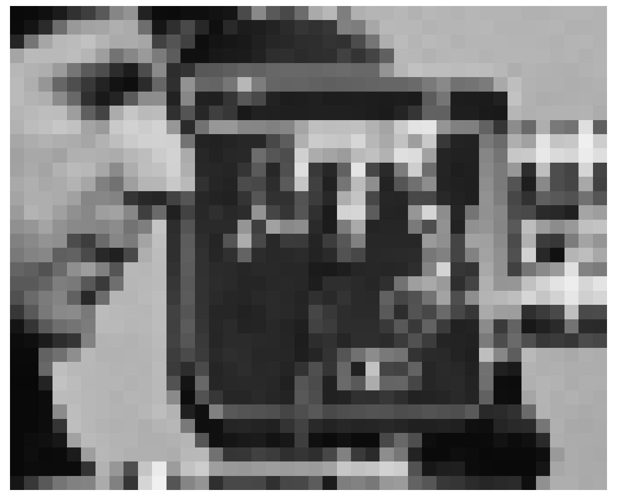
Generalizes [Yuan-Lin '06, Dossal et al. '12, Tibshirani-Taylor '12, ...]

Example: SURE Grid search

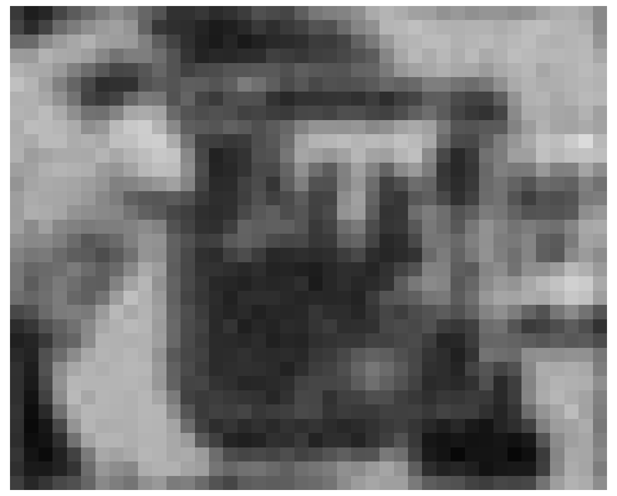
Anisotropic Total Variation

$$J(w) = \|\nabla_{2D} w\|_1$$

X Gaussian convolution

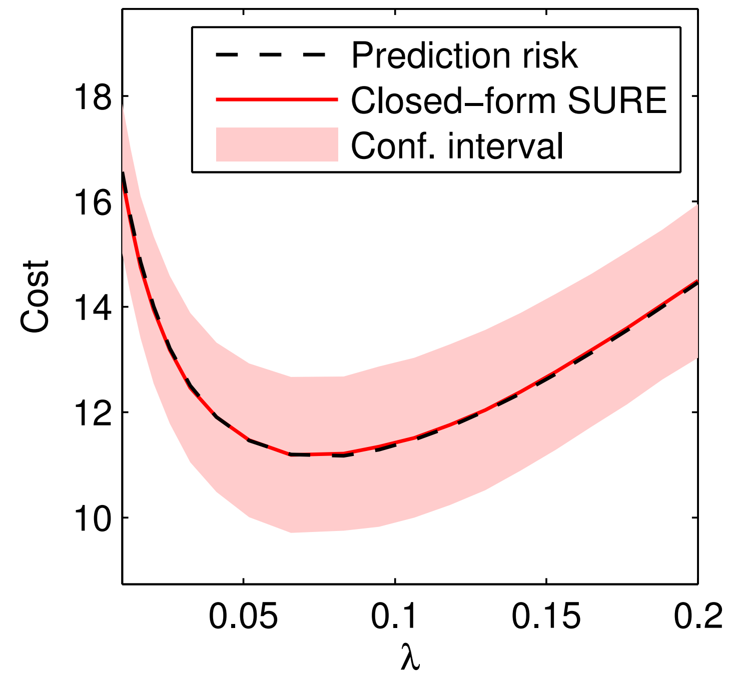


original



blurred

optimal



Explicit SURE: shortcomings

Inverse problems

$y = X w_{\text{true}} + \varepsilon \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$

Prediction

$\hat{\mu}_\theta(y) = X \hat{w}_\theta(y)$

Prediction risk

$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| \hat{\mu}_\theta(y) - X w_{\text{true}} \|_2^2 \right)$

Degrees of freedom and SURE

$$\widehat{\text{df}}_\theta(y) = \text{div}(\hat{\mu}_\theta(y)) = \sum_{i=1}^{n} \frac{\partial (\hat{\mu}_\theta)_i}{\partial y_i}(y)$$

$$\text{SURE}_\theta(y) = \| y - \hat{\mu}_\theta(y) \|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\text{df}}_\theta(y)$$

$$\mathbb{E}_\varepsilon(\text{SURE}_\theta(y)) = \mathcal{R}(\theta)$$

Pb 1: expectation VS realization

only **one** observation $y \implies \mathbb{E}_\varepsilon$ is not computable

(Mostly stable because expectation of a low dimensional quantity with a high dimensional variable)

Pb 2: computational tractability

$\widehat{\text{df}}_\theta(y) = \text{trace}(\text{Jac}_{\hat{\mu}_\theta}(y))$ potentially large

$\longrightarrow$ Monte Carlo SURE

Pb 3: convergence

$w_\theta^{(k)}(y) \xrightarrow{?} \hat{w}_\theta(y)$

stability

$\text{Jac}_{w_\theta^{(k)}}(y) \xrightarrow{?} \text{Jac}_{\hat{w}_\theta}(y)$

very high precision required

Automatic differentiation

J. Bolte, E. Pauwels, SV. Automatic differentiation of non-smooth iterative solvers. 2022

Explicit SURE: shortcomings

Inverse problems

$y = X w_{\text{true}} + \varepsilon \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$

Prediction

$\hat{\mu}_\theta(y) = X \hat{w}_\theta(y)$

Prediction risk

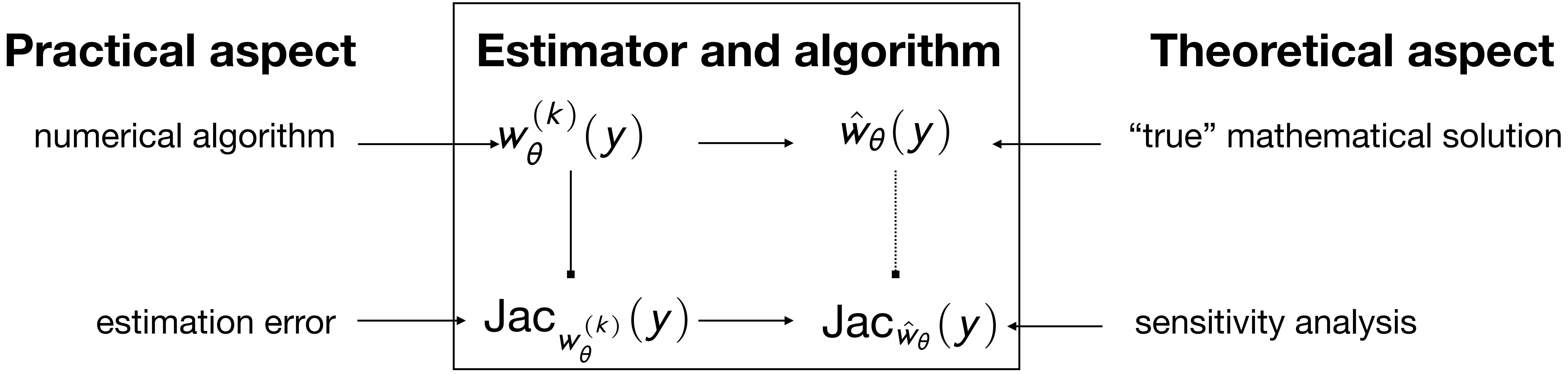
$\mathcal{R}(\theta) = \mathbb{E}_\varepsilon \left(\| \hat{\mu}_\theta(y) - X w_{\text{true}} \|_2^2 \right)$

Degrees of freedom and SURE

$$\widehat{\text{df}}_\theta(y) = \text{div}(\hat{\mu}_\theta(y)) = \sum_{i=1}^{n} \frac{\partial (\hat{\mu}_\theta)_i}{\partial y_i}(y)$$

$$\text{SURE}_\theta(y) = \| y - \hat{\mu}_\theta(y) \|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\text{df}}_\theta(y)$$

$$\mathbb{E}_\varepsilon(\text{SURE}_\theta(y)) = \mathcal{R}(\theta)$$



Convergence of functions does not imply convergence of derivatives!

Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

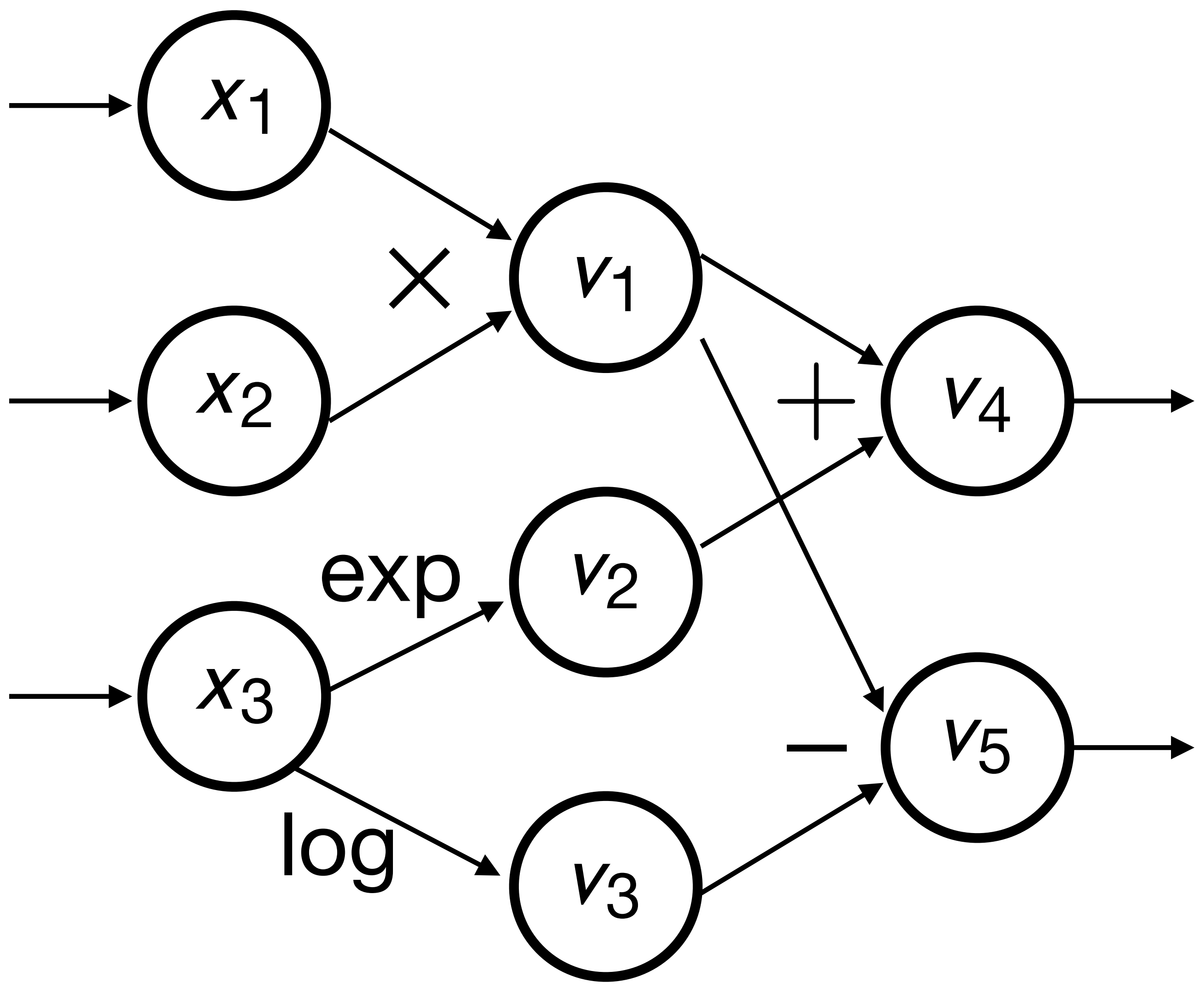
$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

Computer program

```
def f(x1, x2, x3):
    v1 = x1 * x2
    v2 = exp(x3)
    v3 = log(x3)
    v4 = v1 + v2
    v5 = v1 - v3
    return (v4, v5)
```

Forward Tangent Program

Computational graph [Bauer '74]



Inspired by [Baydin '18]

Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

Computer program

1.00 2.00 3.00

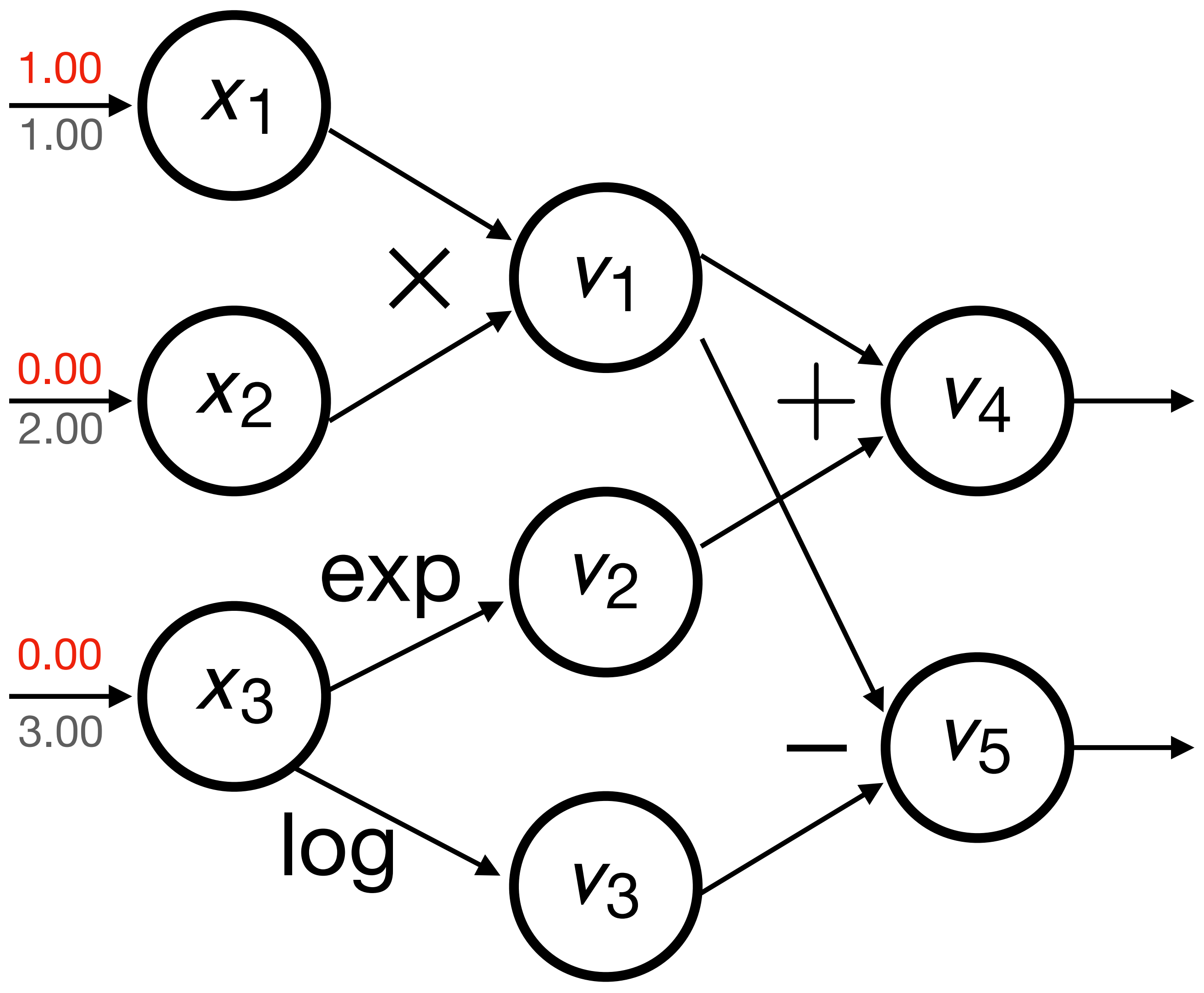
```
def f(x1, x2, x3):
```

Forward Tangent Program

1.00 0.00 0.00

```
def df(dx1, dx2, dx3):
```

Computational graph [Bauer '74]



Inspired by [Baydin '18]

Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\text{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\text{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

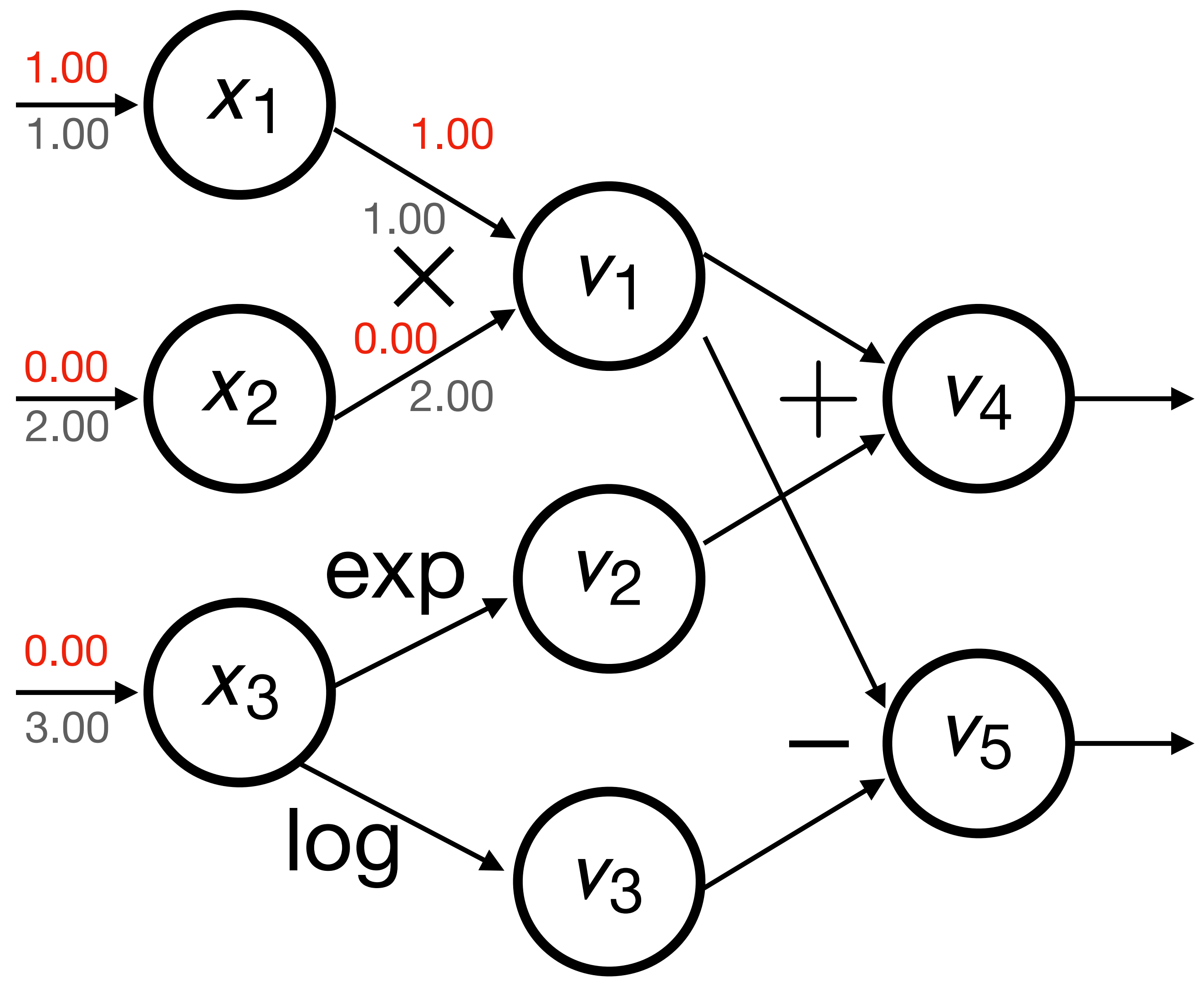
Computer program

```
      1.00  2.00  3.00
def f(x1,   x2,   x3):
    v1 = x1 * x2            2.00
```

Forward Tangent Program

```
       1.00  0.00  0.00
def df(dx1,  dx2,  dx3):
    dv1 = dx1 * x2 + dx1 * x2    2.00
```

Computational graph [Bauer '74]



Inspired by [Baydin '18]

Automatic differentiation: forward mode

Function

$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

Computer program

1.00 2.00 3.00

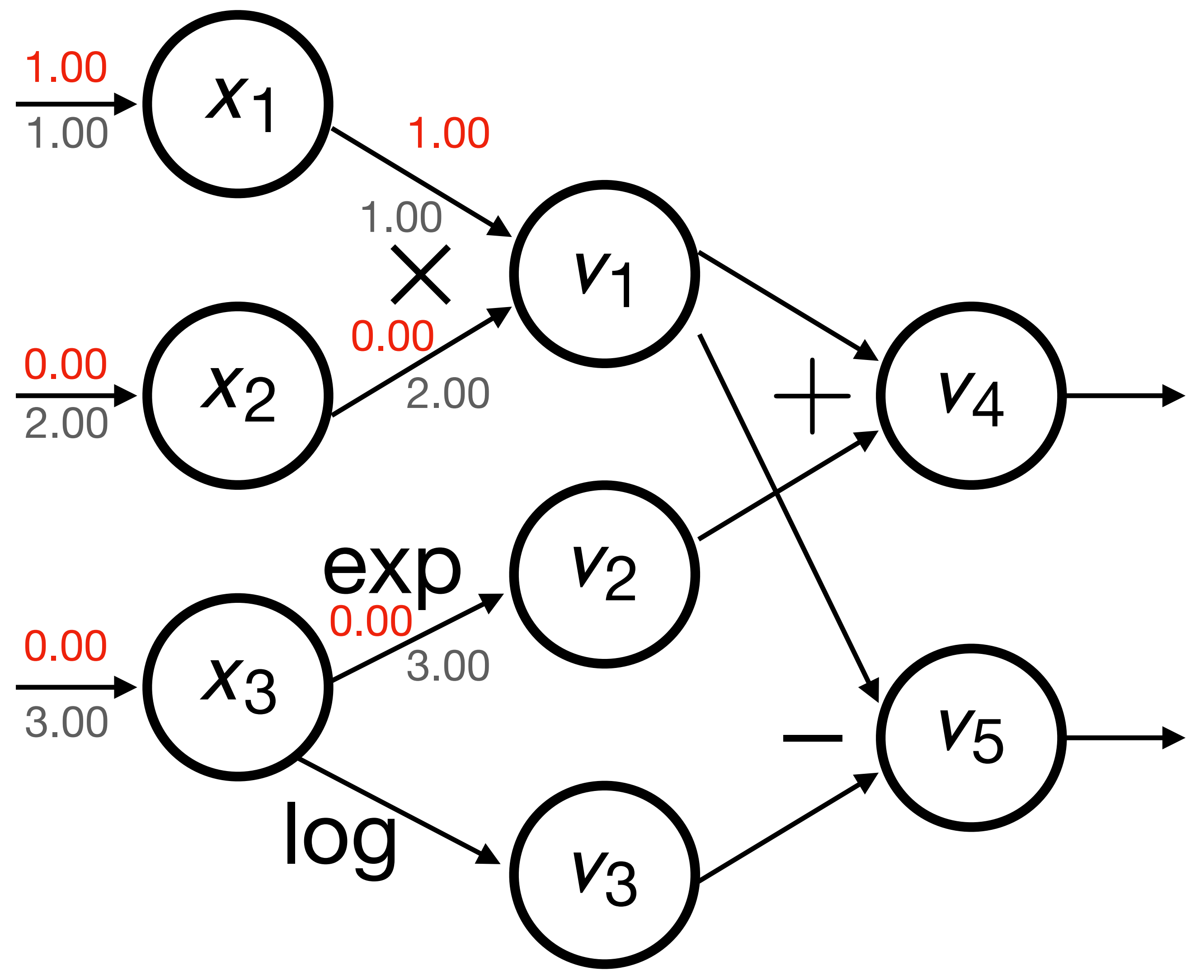
```
def f(x1, x2, x3):
    v1 = x1 * x2          2.00
    v2 = exp(x3)          20.08
```

Forward Tangent Program

1.00 0.00 0.00

```
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
```

Computational graph [Bauer '74]



Inspired by [Baydin '18]

Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

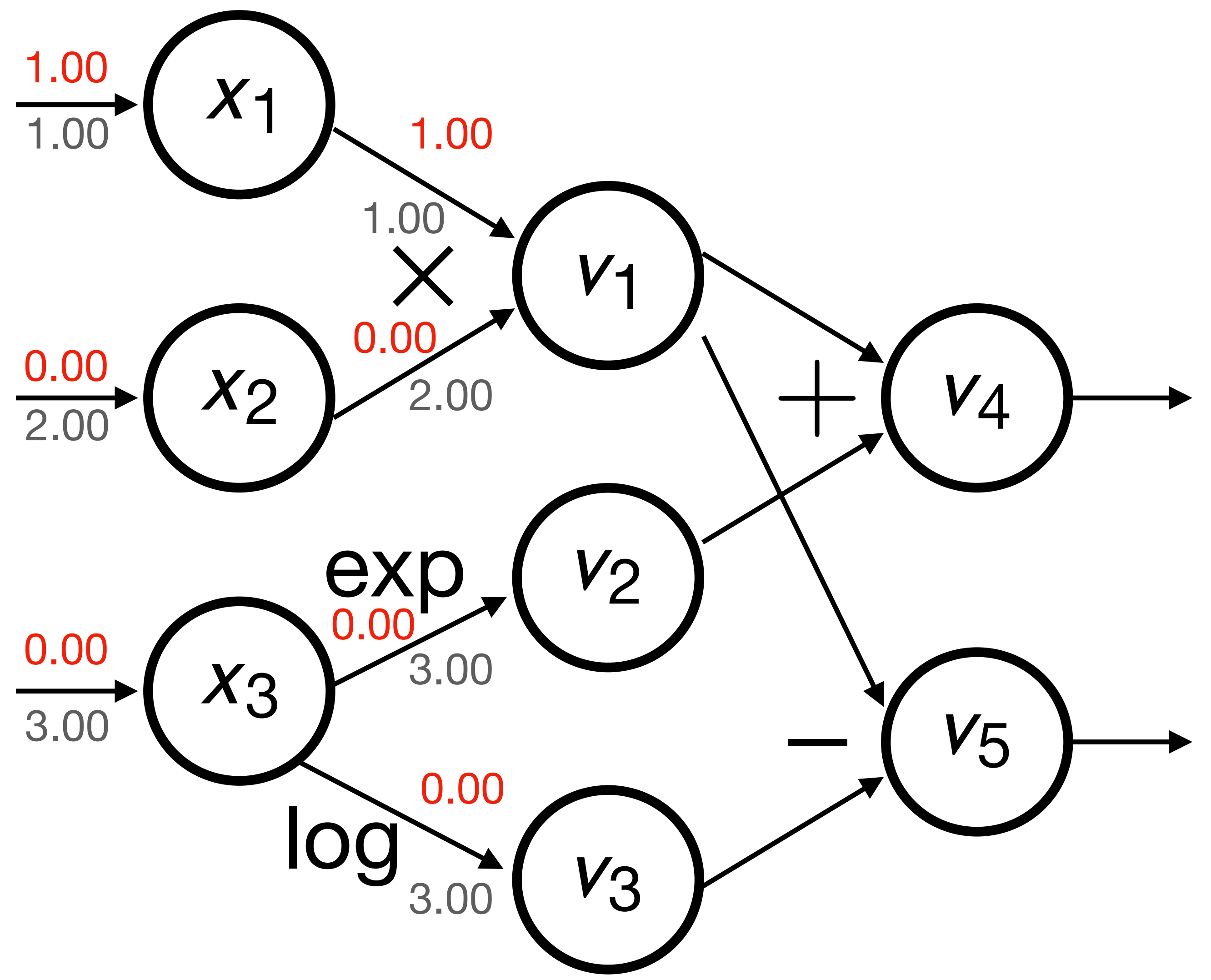
Computer program

```
         1.00  2.00  3.00
def f(x1, x2, x3):
    v1 = x1 * x2          2.00
    v2 = exp(x3)          20.08
    v3 = log(x3)          0.48
```

Forward Tangent Program

```
          1.00  0.00  0.00
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
    dv3 = dx3 / x3                0.00
```

Computational graph [Bauer '74]



Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

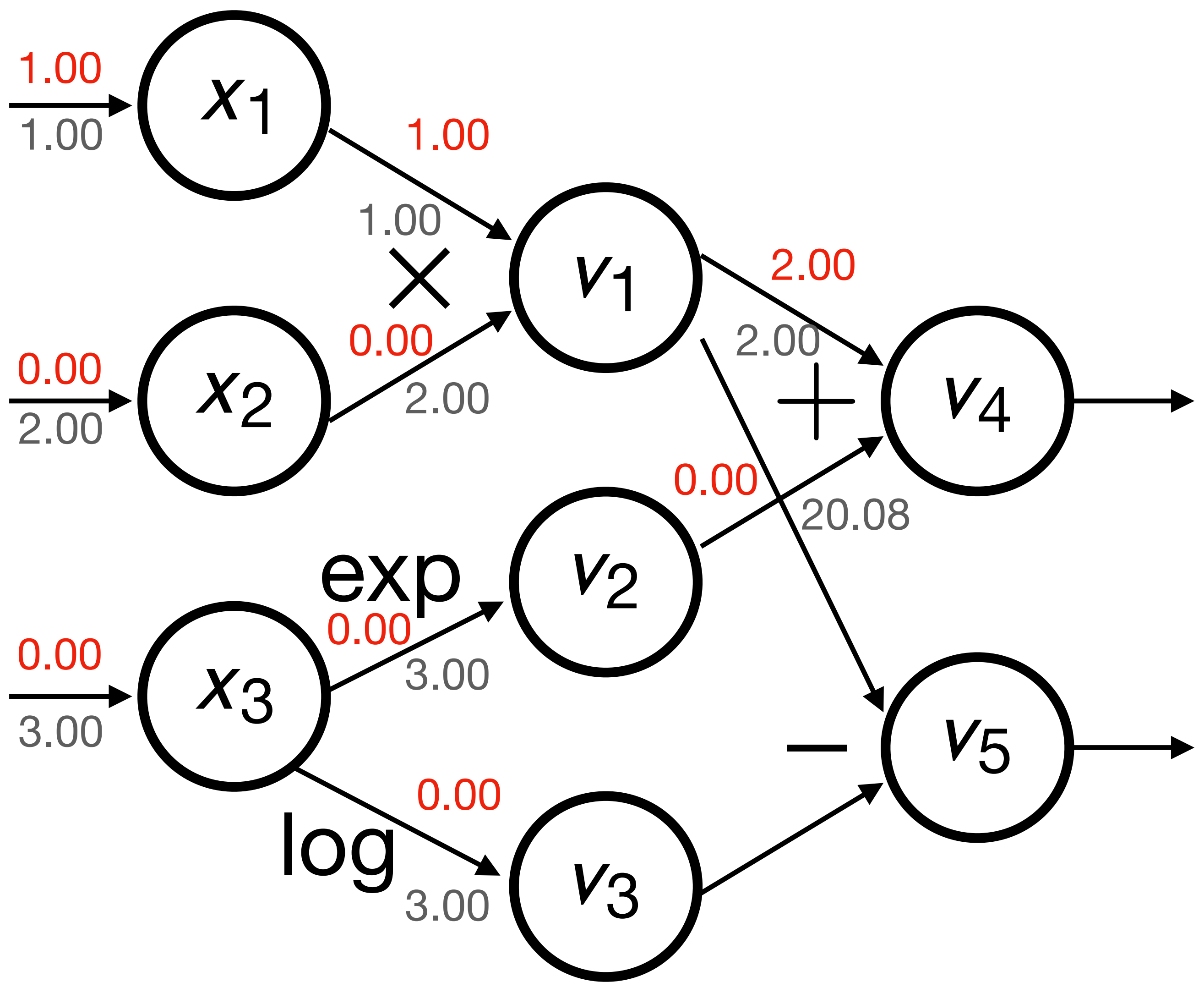
Computer program

```
        1.00  2.00  3.00
def f(x1, x2, x3):
    v1 = x1 * x2          2.00
    v2 = exp(x3)          20.08
    v3 = log(x3)          0.48
    v4 = v1 + v2          22.08
```

Forward Tangent Program

```
         1.00  0.00  0.00
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
    dv3 = dx3 / x3                0.00
    dv4 = dv1 + dv2               2.00
```

Computational graph [Bauer '74]



Inspired by [Baydin '18]

Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\text{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\text{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

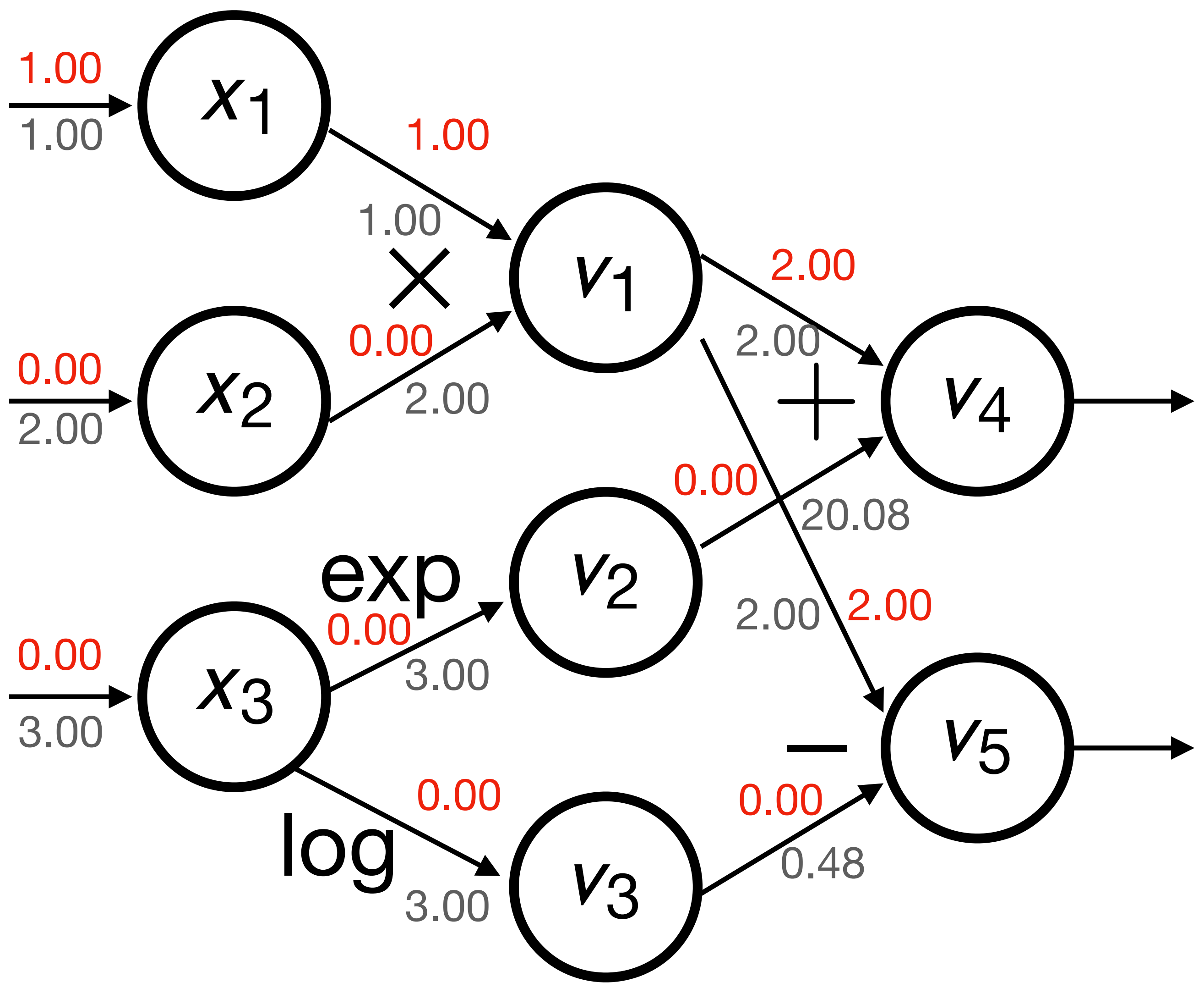
Computer program

```
      1.00  2.00  3.00
def f(x1, x2, x3):
    v1 = x1 * x2        2.00
    v2 = exp(x3)        20.08
    v3 = log(x3)        0.48
    v4 = v1 + v2        22.08
    v5 = v1 - v3        2.08
```

Forward Tangent Program

```
       1.00  0.00  0.00
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
    dv3 = dx3 / x3                0.00
    dv4 = dv1 + dv2               2.00
    dv5 = dv1 - dv3               2.00
```

Computational graph [Bauer '74]



Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \longrightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\mathrm{Jac}_f : \mathbb{R}^p \longrightarrow \mathbb{R}^{n \times p}$$

$$\mathrm{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

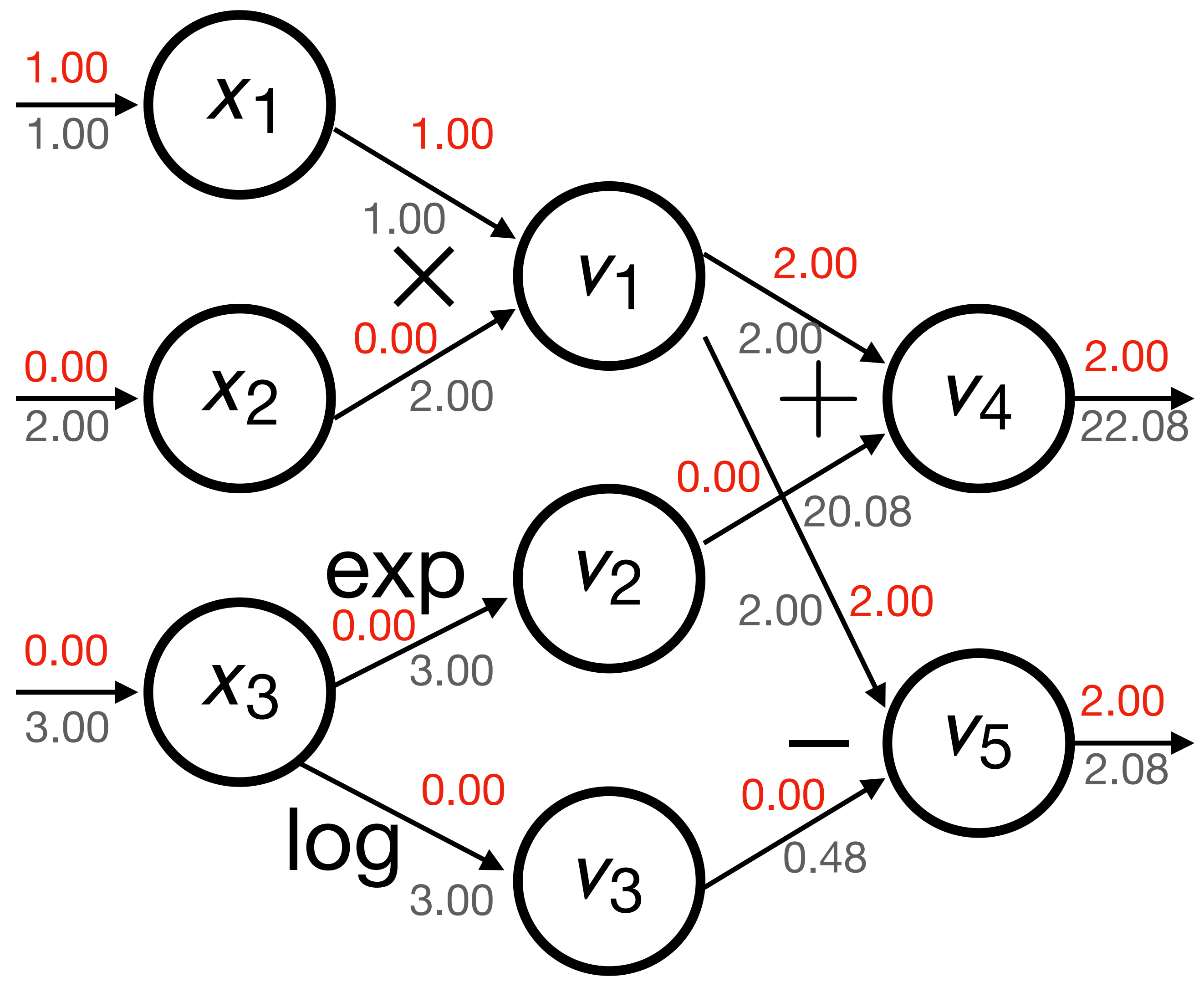
Computer program

```
        1.00   2.00   3.00
def f(x1, x2, x3):
    v1 = x1 * x2              2.00
    v2 = exp(x3)              20.08
    v3 = log(x3)              0.48
    v4 = v1 + v2              22.08
    v5 = v1 - v3              2.08
    return (v4, v5)  (22.08, 2.08)
```

Forward Tangent Program

```
         1.00   0.00   0.00
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
    dv3 = dx3 / x3                0.00
    dv4 = dv1 + dv2               2.00
    dv5 = dv1 - dv3               2.00
    return (dv4, dv5)      (2.00, 2.00)
```

Computational graph [Bauer '74]



Automatic differentiation: forward mode

Function

$$f : \mathbb{R}^p \rightarrow \mathbb{R}^n$$

$$f(x_1, x_2, x_3) = \begin{pmatrix} x_1 x_2 + \exp(x_3) \\ x_1 x_2 - \log(x_1) \end{pmatrix}$$

Jacobian

$$\text{Jac}_f : \mathbb{R}^p \rightarrow \mathbb{R}^{n \times p}$$

$$\text{Jac}_f(x_1, x_2, x_3) = \begin{pmatrix} x_2 & x_1 & \exp(x_3) \\ x_2 & x_1 & 1/x_3 \end{pmatrix}$$

$$\frac{\partial f}{\partial x_1}(1, 2, 3) = (2, 2)^\top$$

- Computing the full Jacobian requires p calls to tangent program
- No memory requirement
- Performant when $p < n$

[Wengert '64, Griewank '89]

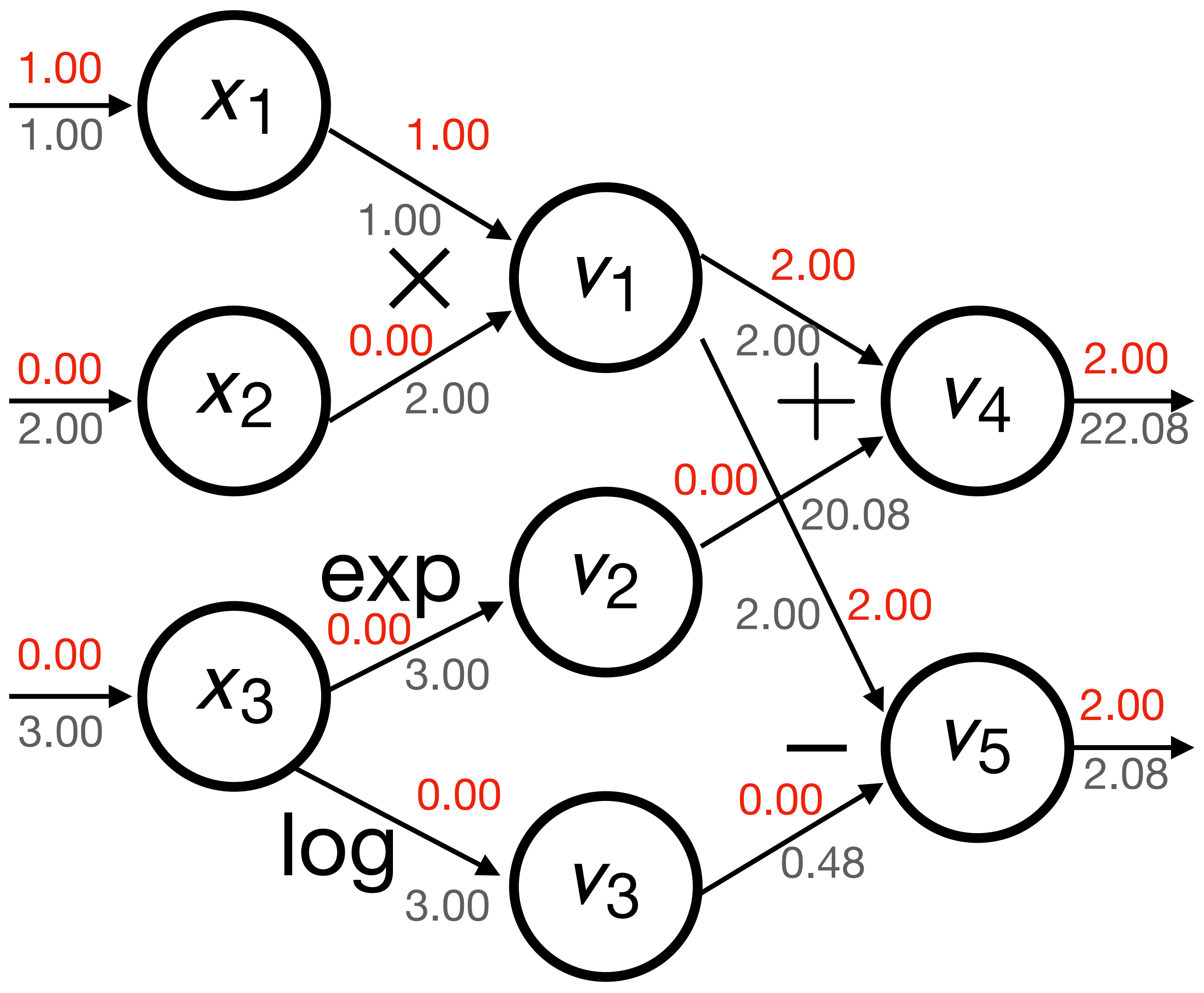
Computer program

```
       1.00  2.00  3.00
def f(x1, x2, x3):
    v1 = x1 * x2            2.00
    v2 = exp(x3)            20.08
    v3 = log(x3)            0.48
    v4 = v1 + v2            22.08
    v5 = v1 - v3            2.08
    return (v4, v5)         (22.08, 2.08)
```

Forward Tangent Program

```
        1.00  0.00  0.00
def df(dx1, dx2, dx3):
    dv1 = dx1 * x2 + dx1 * x2     2.00
    dv2 = dx3 * exp(x3)           0.00
    dv3 = dx3 / x3                0.00
    dv4 = dv1 + dv2               2.00
    dv5 = dv1 - dv3               2.00
    return (dv4, dv5)             (2.00, 2.00)
```

Computational graph [Bauer '74]



Forward automatic differentiation

=

Forward derivative accumulation

=

Tangent linear mode

=

Line-to-line derivation

=

Jacobian-vector product

$\text{Jac}_f(x) \cdot z$

$\neq$

~~Backpropagation~~

Note that in practice the forward tangent program need to be implemented in the same function (variables accessibility)

Inspired by [Baydin '18]

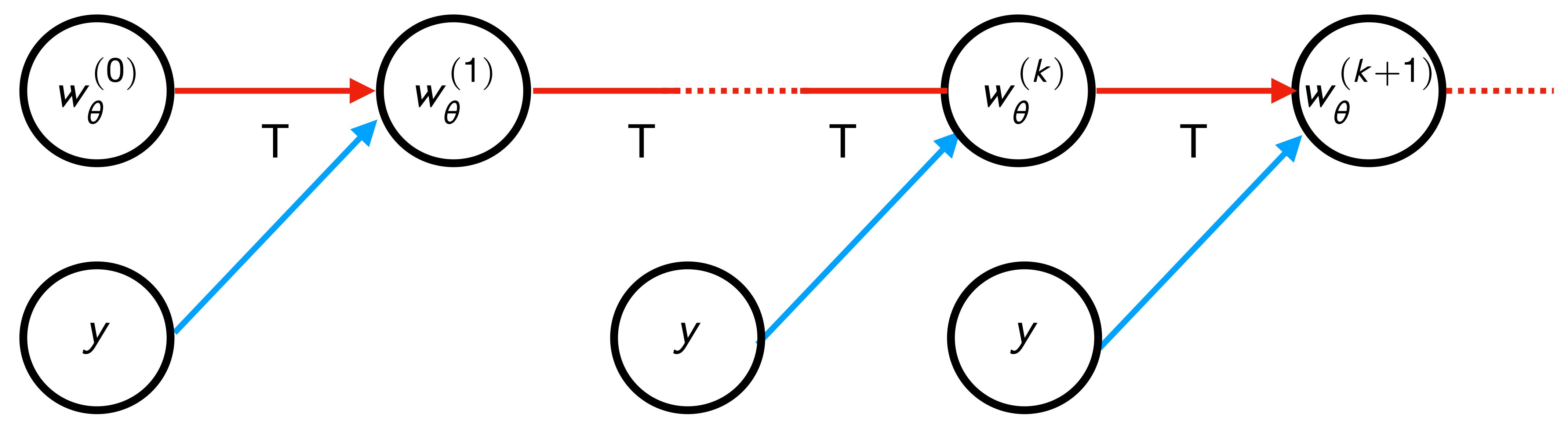
Estimating a Jacobian of iterative estimator

Iterative estimators $w_\theta^{(k)}(y)$

$$\mathrm{T} : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^p$$

$$\begin{cases} w_\theta^{(k)}(y) \xrightarrow{k \to +\infty} \hat{w}_\theta(y) \\ w_\theta^{(k+1)}(y) = \mathrm{T}(w_\theta^{(k)}(y), y) \end{cases}$$

Computational graph



Chain rule

$$\mathrm{Jac}_{w_\theta^{k+1}}(y) = \partial_1 \mathrm{T}(w_\theta^k(y), y) \cdot \mathrm{Jac}_{w_\theta^k(y)}(y) + \partial_2 \mathrm{T}(w_\theta^k(y), y)$$

$p \times n$ $\quad$ $p \times p$ $\quad$ $p \times n$ $\quad$ $p \times n$

Forward differentiation "Jacobian-vector product"

$$\begin{cases} w_\theta^{(k+1)}(y) & = \mathrm{T}(w_\theta^{(k)}(y), y) \\ \mathrm{d}w_\theta^{(k+1)}(y) & = \partial_1 \mathrm{T}(w_\theta^k(y), y) \cdot \mathrm{d}w_\theta^{(k+1)}(y) + \partial_2 \mathrm{T}(w_\theta^k(y), y)\delta \end{cases}$$

Well-foundness of iterative differentiation

Iterative estimators $w_\theta^{(k)}(y)$

$$\mathrm{T} : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^p$$

$$\begin{cases} w_\theta^{(k)}(y) \xrightarrow{k \to +\infty} \hat{w}_\theta(y) \\ w_\theta^{(k+1)}(y) = \mathrm{T}(w_\theta^{(k)}(y), y) \end{cases}$$

Forward differentiation "Jacobian-vector product"

$$w_\theta^{(k+1)}(y) = \mathrm{T}(w_\theta^{(k)}(y), y)$$

$$\mathrm{d}w_\theta^{(k+1)}(y) = \partial_1 \mathrm{T}(w_\theta^k(y), y) \cdot \mathrm{d}w_\theta^{(k+1)}(y) + \partial_2 \mathrm{T}(w_\theta^k(y), y)\delta$$

Theorem [Bolte, Pauwels, V '22]

If T is loc. Lipschitz and its *conservative Jacobian* is a contraction, then for almost all y,

$$\lim_{k \to +\infty} \mathrm{d}w_\theta^{(k)}(y) = \frac{\partial \hat{w}_\theta(y)}{\partial y} \cdot \delta$$

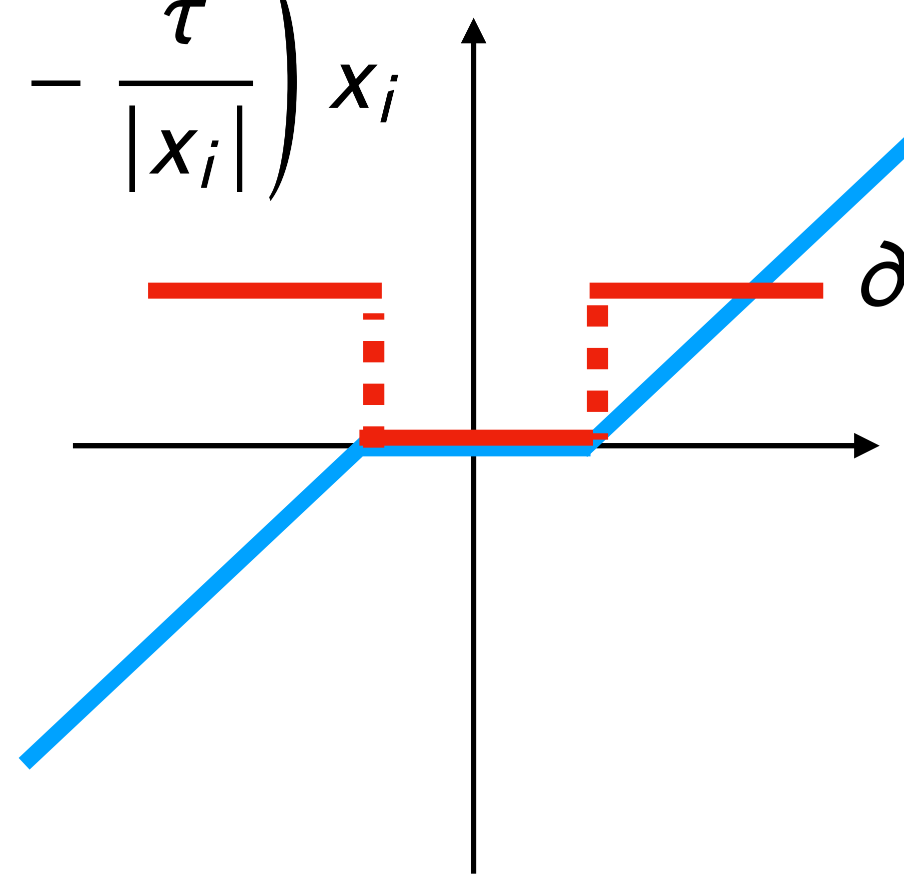
Forward differentiation of a Lasso solver

Lasso

$$\hat{w}_\theta(y) \in \underset{w \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2}\|y - Xw\|_2^2 + \theta\|w\|_1$$

Soft-thresholding

$$\operatorname{prox}_{\tau\|\cdot\|_1}(x)_i = \min\left(0, 1 - \frac{\tau}{|x_i|}\right) x_i$$



Iterative soft-thresholding (Forward-Backward)

$$w_\theta^{(k+1)}(y) = \operatorname{prox}_{\tau\theta\|\cdot\|_1}(w_\theta^{(k)}(y) - \tau X^\top(Xw_\theta^{(k)}(y) - y))$$

If $\tau < 2/\|X\|^2$, $w_\theta^{(k)}(y) \to \hat{w}_\theta(y)$

Derivative of Forward-Backward

$$\begin{aligned}
\partial_1 \mathrm{T}(w, y) &= \mathrm{Jac}_{\operatorname{prox}_{\tau\theta\|\cdot\|_1}}(w - \tau X^\top(Xw - y)) \cdot \mathrm{Jac}_{w \mapsto w - \tau X^\top(Xw - y)}(w) \\
&= \partial\operatorname{prox}_{\tau\theta\|\cdot\|_1}(w - \tau X^\top(Xw - y)) \cdot (\mathrm{Id} - \tau X^\top X) \\
\partial_2 \mathrm{T}(w, y) &= \mathrm{Jac}_{\operatorname{prox}_{\tau\theta\|\cdot\|_1}}(w - \tau X^\top(Xw - y)) \cdot \mathrm{Jac}_{y \mapsto w - \tau X^\top(Xw - y)}(y) \\
&= \partial\operatorname{prox}_{\tau\theta\|\cdot\|_1}(w - \tau X^\top(Xw - y)) \cdot \tau X
\end{aligned}$$

The SUGAR way

B. Pascal, SV, N. Pustelnik, P. Abry. Automated data-driven selection of the hyperparameters for Total-Variation based texture segmentation. 2020.
C. Deledalle, SV, J. Fadili, G. Peyré. Stein Unbiased GrAdient estimator of the Risk (SUGAR) for multiple parameter selection. SIAM J Imaging Sci. 7(4):2448—2487. 2014.
C. Deledalle, SV, G. Peyré, J. Fadili, C. Dossal. Proximal Splitting Derivatives for Risk Estimation. *NCMIP*. 2012.

Fifty shades of SURE

Closed-form SURE [Stein '81]

$$\mathrm{SURE}_\theta(y) = \|y - \hat{\mu}_\theta(y)\|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\mathrm{df}}_\theta(y) \qquad \widehat{\mathrm{df}}_\theta(y) = \mathrm{trace}(\mathrm{Jac}_{\hat{\mu}_\theta}(y))$$

Issue: dimensionality of the Jacobian

$$\mathbb{E}_z\left(\widehat{\mathrm{df}}_\theta^{\mathrm{MC}}(y)\right) = \widehat{\mathrm{df}}_\theta(y)$$

Monte-Carlo SURE [Ramani et al. '08]

$$z \sim \mathcal{N}(0, \mathrm{Id})$$

$$\mathrm{SURE}_\theta^{\mathrm{MC}}(y) = \|y - \hat{\mu}_\theta(y)\|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\mathrm{df}}_\theta^{\mathrm{MC}}(y) \qquad \widehat{\mathrm{df}}_\theta^{\mathrm{MC}}(y) = \langle \mathrm{Jac}_{\hat{\mu}_\theta}(y) z, \, z \rangle$$

Issue: accessibility of the true Jacobian

need proof
of convergence

Iterative Monte-Carlo SURE [Vonesch et al. '08, Giryes et al. '11, Ramani et al. '12, Deledalle et al. '12]

$$\mathrm{SURE}_\theta^{(k),\mathrm{MC}}(y) = \|y - \mu_\theta^{(k)}(y)\|_2^2 - n\sigma^2 + 2\sigma^2 \widehat{\mathrm{df}}_\theta^{(k),\mathrm{MC}}(y) \qquad \widehat{\mathrm{df}}_\theta^{(k),\mathrm{MC}}(y) = \langle \mathrm{d}w_\theta^{(k)}, \, z \rangle$$

Fifty shades of SURE — continued

DoF, Monte-Carlo DoF and Iterative Monte-Carlo DoF

$$\widehat{\mathrm{df}}_\theta(y) = \mathrm{trace}(\mathrm{Jac}_{\hat\mu_\theta}(y)) \qquad \widehat{\mathrm{df}}_\theta^{\mathrm{MC}}(y) = \langle \mathrm{Jac}_{\hat\mu_\theta}(y) z,\ z\rangle \qquad \widehat{\mathrm{df}}_\theta^{(k),\mathrm{MC}}(y) = \langle \mathrm{d}w_\theta^{(k)},\ z\rangle$$

Finite-difference SURE [Ye '98, Shen-Ye '02, Ramani et al. '08]

$$\widehat{\mathrm{df}}_\theta^{\mathrm{FD}}(y) = \frac{1}{\delta}\sum_{i=1}^{n} (\hat\mu_\theta(y+\delta \mathbf{e}_i) - \hat\mu_\theta(y))_i$$

If $\hat\mu_\theta$ Lipschitz-continuous

$$\lim_{\delta\to 0} \widehat{\mathrm{df}}_\theta^{\mathrm{FD}}(y) = \widehat{\mathrm{df}}_\theta(y)$$

Issues:

- numerical instabilities
- observation dimension

Finite-difference Monte-Carlo SURE

$$\widehat{\mathrm{df}}_\theta^{\mathrm{FDMC}}(y) = \frac{1}{\delta}\langle \hat\mu_\theta(y+\delta z) - \hat\mu_\theta(y),\ z\rangle$$

$$\mathbb{E}_z\left(\widehat{\mathrm{df}}_\theta^{\mathrm{FDMC}}(y)\right) = \widehat{\mathrm{df}}_\theta^{\mathrm{FD}}(y)$$

Issue: accessibility of the true estimator!

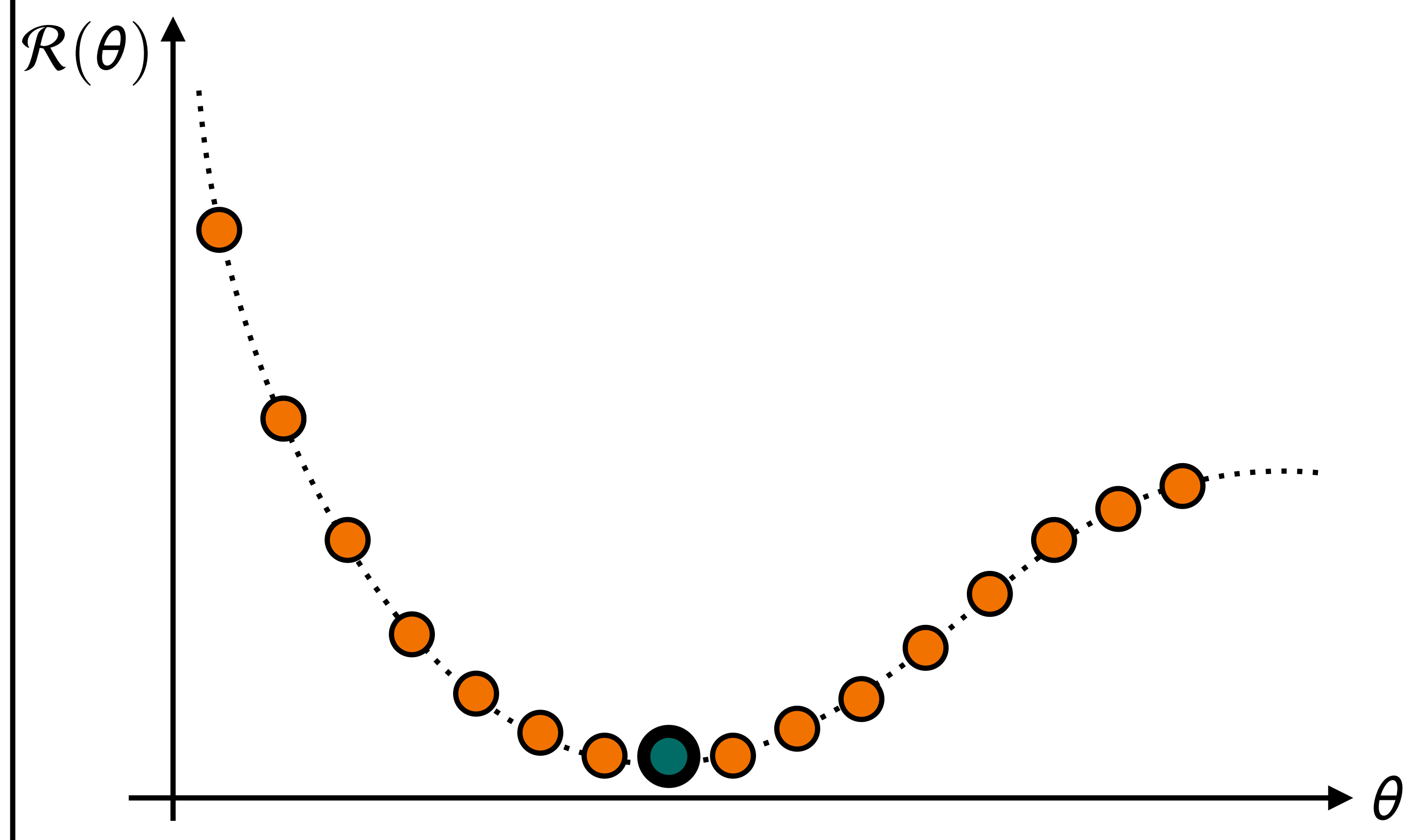
Iterative Finite-difference Monte-Carlo SURE

$$\widehat{\mathrm{df}}_\theta^{(k),\mathrm{FDMC}}(y) = \tfrac{1}{\delta}\langle \mu_\theta^{(k)}(y+\delta z) - \mu_\theta^{(k)}(y),\ z\rangle$$

Reminder: 0-order vs 1-order search

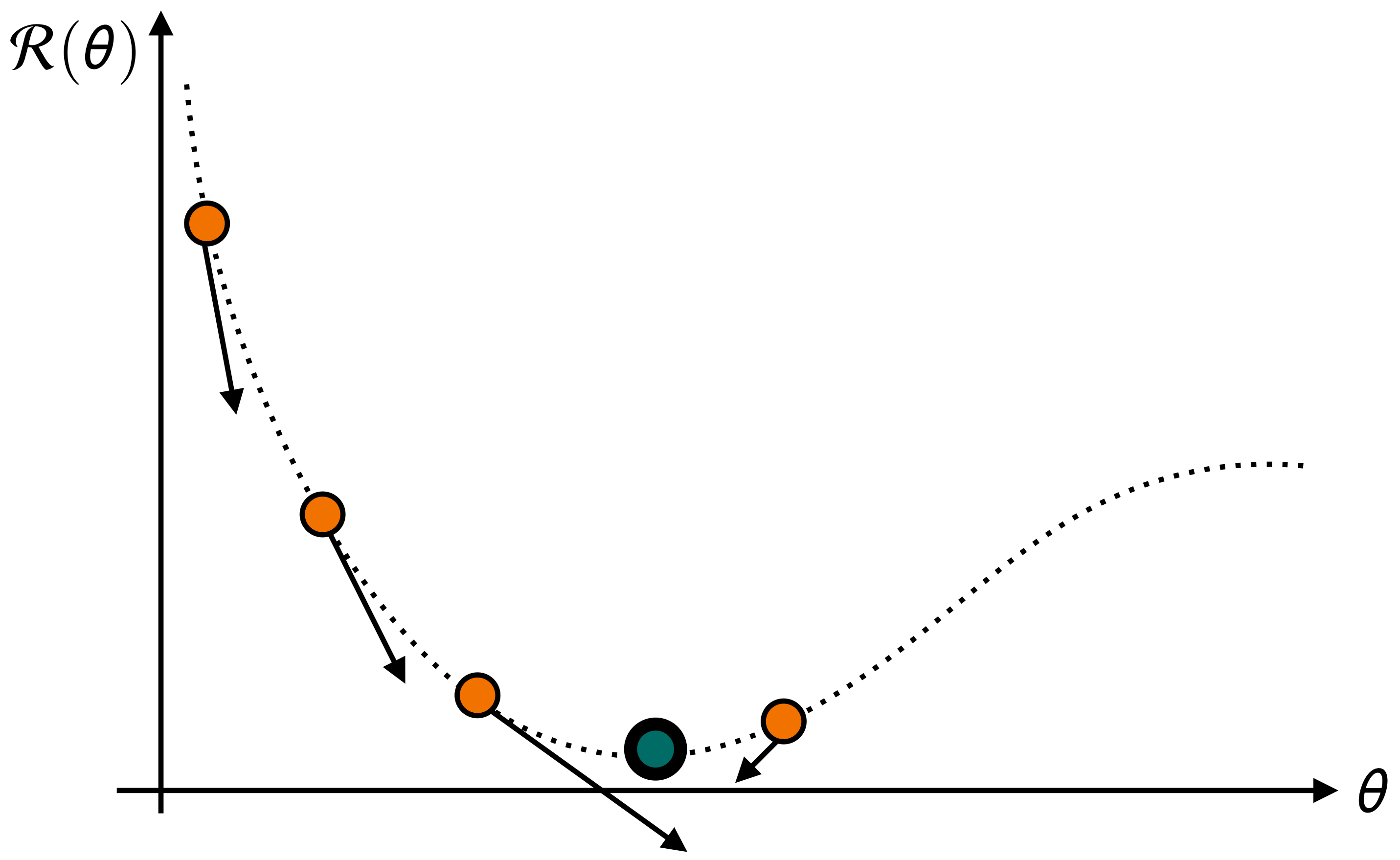
Grid search

1. Choose a criterion $\mathcal{R}$
2. Sample uniformly Θ
3. Evaluate $\mathcal{R}(\theta)$ on the grid
4. Keep the best $\theta^\star$



Hyper-gradient descent

1. Choose a criterion $\mathcal{R}$
2. Take a starting point
3. Eval. $\mathcal{R}(\theta), \nabla\mathcal{R}(\theta)$ on-the-fly
4. Keep the last iterate $\theta^\star$



BUT

$$\theta \mapsto \mathbb{E}_\varepsilon\left(\|\hat{\mu}_\theta(y) - X w_{\text{true}}\|_2^2\right)$$

(weakly) differentiable

$$\not\Downarrow$$

$$\theta \mapsto \mathrm{SURE}_\theta(y)$$

differentiable or continuous

What about $\mathrm{SURE}_\theta^{\mathrm{FD}}(y)$, $\mathrm{SURE}_\theta^{\mathrm{FDMC}}(y)$?

From SURE to SUGAR

$$\widehat{\mathrm{df}}_\theta^{\mathrm{FD}}(y) = \frac{1}{\delta}\sum_{i=1}^{n}\left(\hat{\mu}_\theta(y+\delta\mathbf{e}_i) - \hat{\mu}_\theta(y)\right)_i \qquad \widehat{\mathrm{df}}_\theta^{\mathrm{FDMC}}(y) = \frac{1}{\delta}\langle \hat{\mu}_\theta(y+\delta z) - \hat{\mu}_\theta(y),\, z\rangle$$

Proposition [Deledalle et al. '14, Pascal et al. '20]

Assume $(y,\theta) \mapsto \hat{\mu}_\theta(y)$ weakly differentiable *w.r.t* y and θ

Given $\delta > 0$, $\widehat{\mathrm{df}}_\theta^{\mathrm{FD}}(y)$, $\widehat{\mathrm{df}}_\theta^{\mathrm{FDMC}}(y)$ are also weakly differentiable

$\mathrm{SURE}_\theta^{\mathrm{FD}}(y)$, $\mathrm{SURE}_\theta^{\mathrm{FDMC}}(y)$

Stein Unbiased GrAdient estimator of the Risk (SUGAR)

$$\mathrm{SUGAR}_\theta^{\mathrm{FD/FDMC}}(y) = \nabla_\theta\left(\mathrm{SURE}_\bullet^{\mathrm{FD/FDMC}}(\bullet)\right)(y,\theta)$$

$$= 2\mathrm{Jac}_{\hat{\mu}_\bullet(\theta)}(y)^\top(\hat{\mu}_\theta(y) - y) + 2\sigma^2\nabla_\theta\left(\widehat{\mathrm{df}}_\bullet^{\mathrm{FD/FDMC}}(\bullet)\right)(y,\theta)$$

$$\nabla_\theta\left(\widehat{\mathrm{df}}_\bullet^{\mathrm{FD}}(\bullet)\right)(y,\theta) = \frac{1}{\delta}\sum_{i=1}^{n}\left(\mathrm{Jac}_{\hat{\mu}_\bullet(y+\delta\mathbf{e}_i)}(\theta) - \mathrm{Jac}_{\hat{\mu}_\bullet(y)}(\theta)\right)^\top \mathbf{e}_i$$

$$\nabla_\theta\left(\widehat{\mathrm{df}}_\bullet^{\mathrm{FDMC}}(\bullet)\right)(y,\theta) = \frac{1}{\delta}\left(\mathrm{Jac}_{\hat{\mu}_\bullet(y+\delta z)}(\theta) - \mathrm{Jac}_{\hat{\mu}_\bullet(y)}(\theta)\right)^\top z$$

Asymptotic unbiasedness of SUGAR

Theorem [Deledalle et al. '14, Pascal et al. '20]

Assume $(y, \theta) \mapsto \hat{\mu}_\theta(y)$ Lipschitz-continuous *w.r.t* y and θ

$$\lim_{\delta \to 0} \mathbb{E}_\varepsilon \left(\mathrm{SUGAR}_\theta^{\mathrm{FD/FDMC}}(y) \right) = \nabla_\theta \mathbb{E}_\varepsilon \left(\|\hat{\mu}_\bullet(y) - X w_{\mathrm{true}}\|_2^2 \right)(\theta)$$

In practice, δ / n need to not decrease quickly to ensure numerical stability

Stein Unbiased GrAdient estimator of the Risk (SUGAR)

$$\begin{aligned} \mathrm{SUGAR}_\theta^{\mathrm{FD/FDMC}}(y) &= \nabla_\theta (\mathrm{SURE}_\bullet{}^{\mathrm{FD/FDMC}}(\bullet))(y, \theta) \\ &= 2 \mathrm{Jac}_{\hat{\mu}_\bullet(\theta)}(y)^\top (\hat{\mu}_\theta(y) - y) + 2\sigma^2 \nabla_\theta (\widehat{\mathrm{df}}_\bullet{}^{\mathrm{FD/FDMC}}(\bullet))(y, \theta) \end{aligned}$$

$$\nabla_\theta (\widehat{\mathrm{df}}_\bullet{}^{\mathrm{FDMC}}(\bullet))(y, \theta) = \frac{1}{\delta} \left(\mathrm{Jac}_{\hat{\mu}_\bullet(y + \delta z)}(\theta) - \mathrm{Jac}_{\hat{\mu}_\bullet(y)}(\theta) \right)^\top z$$

SUGAR for iterative estimators

Iterative estimators $w_\theta^{(k)}(y)$

$$\mathrm{T} : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^p$$

$$\begin{cases} w_\theta^{(k+1)}(y) & = \mathrm{T}(w_\theta^{(k)}(y), y) \\ \mathrm{d}w_\theta^{(k+1)}(y) & = \partial_1 \mathrm{T}(w_\theta^k(y), y) \cdot \mathrm{d}w_\theta^{(k)}(y) + \partial_2 \mathrm{T}(w_\theta^k(y), y) z \end{cases}$$

Running time x2

Global complexity: **4x** original iterative estimator

Iterative SUGAR Finite-difference + Monte-Carlo

$$\mathrm{SUGAR}_\theta^{(k),\mathrm{FD/FDMC}}(y) = \nabla_\theta(\mathrm{SURE}_\bullet{}^{(k),\mathrm{FD/FDMC}}(\bullet))(y, \theta)$$

$$= 2\mathrm{Jac}_{\mu_\bullet{}^{(k)}(\theta)}(y)^\top(\mu_\theta^{(k)}(y) - y) + 2\sigma^2 \nabla_\theta(\widehat{\mathrm{df}}_\bullet{}^{(k),\mathrm{FD/FDMC}}(\bullet))(y, \theta)$$

$$\nabla_\theta(\widehat{\mathrm{df}}_\bullet{}^{(k),\mathrm{FDMC}}(\bullet))(y, \theta) = \frac{1}{\delta}(\mathrm{d}w_\theta{}^{(k)}(y + \delta z) - \mathrm{d}w_\theta{}^{(k)}(y))$$

Running time x2

SUGAR in practice: image deblurring

Total variation deblurring

$$y = X w_{\text{true}} + \varepsilon$$

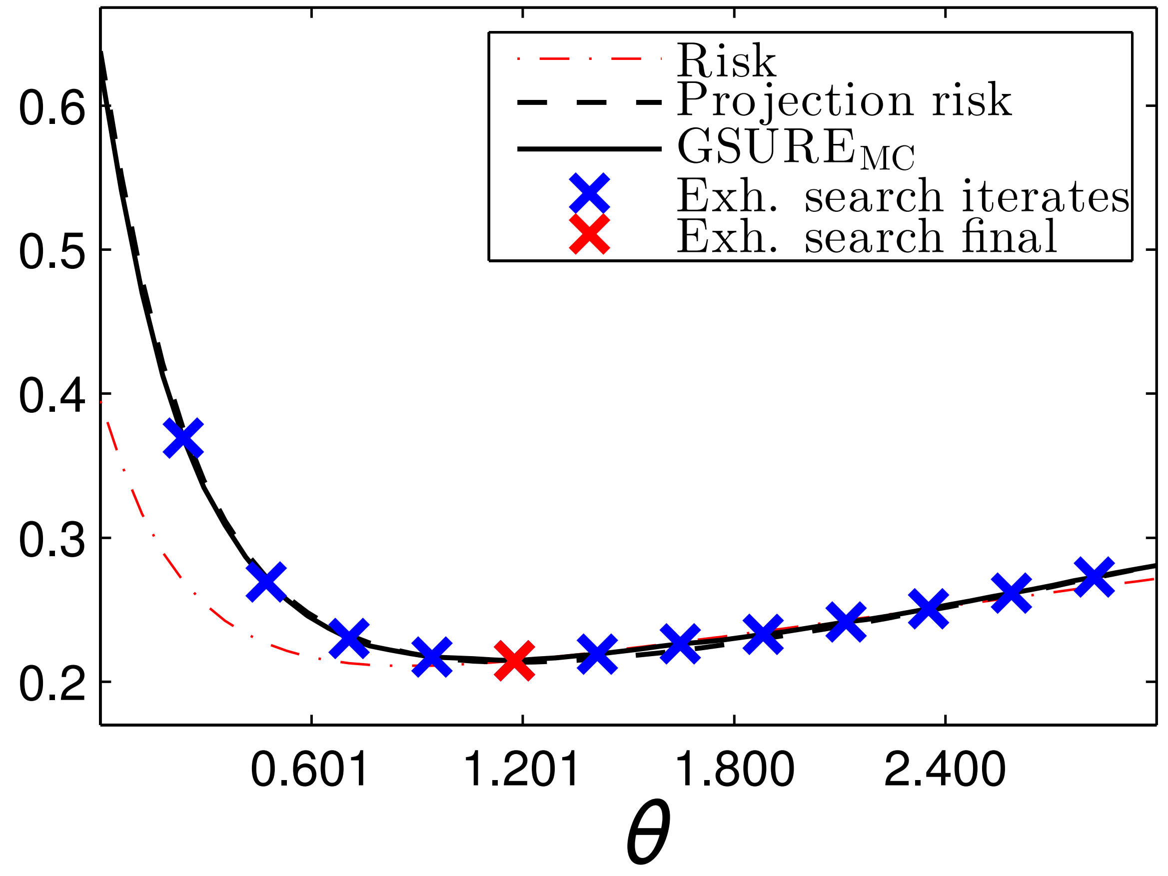
$$\hat{w}_\theta(y) \in \underset{w}{\operatorname{argmin}} \frac{1}{2}\|y - X w_{\text{true}}\|_2^2 + \theta \|\nabla_{2D} w\|_{1,2}$$

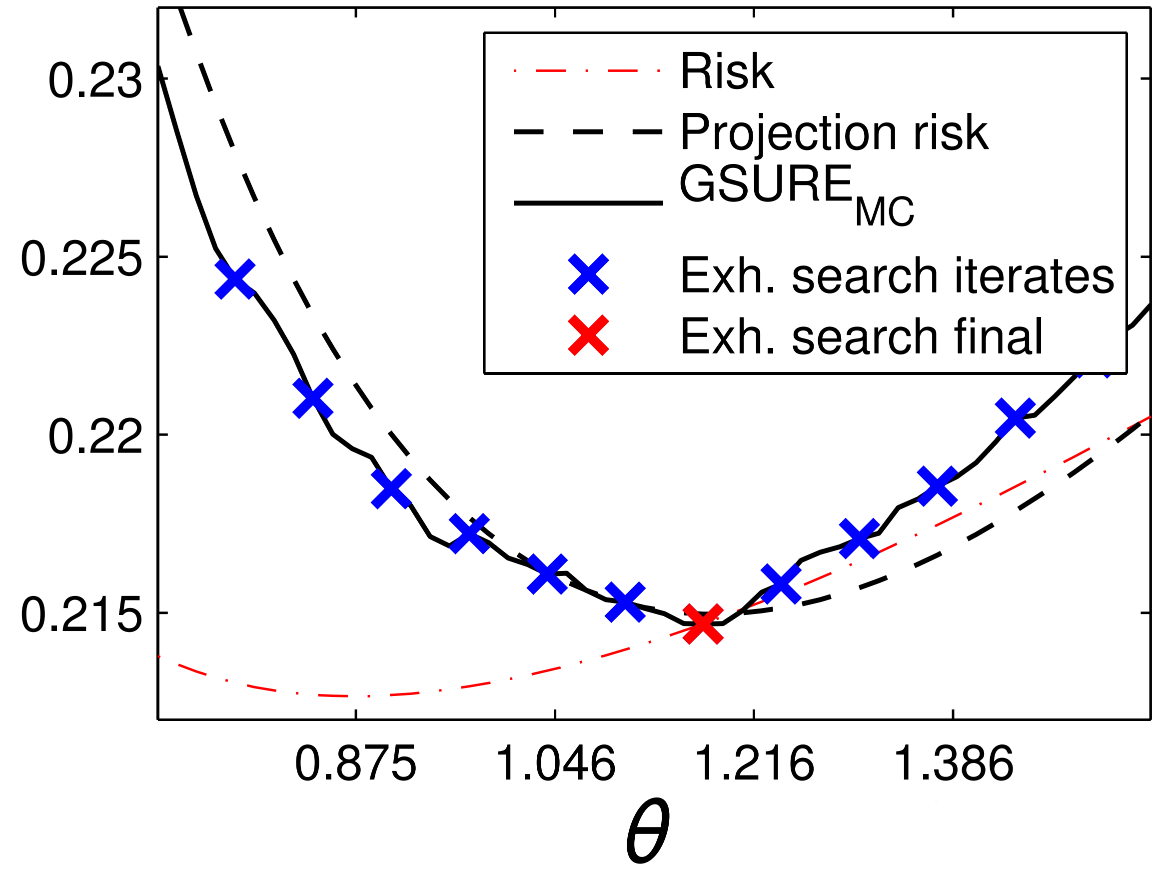
[Chambolle-Pock ’11]

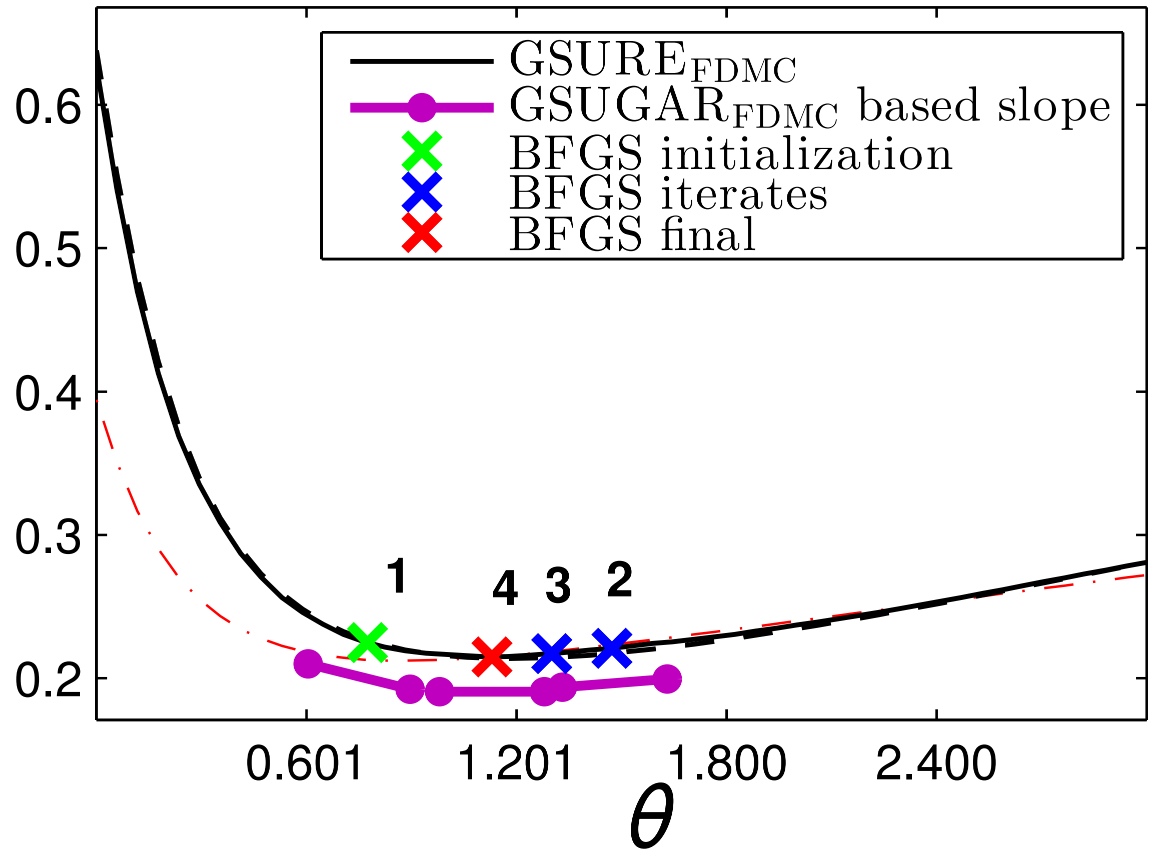
$w_\theta^{(k)}(y)$ computed with a differentiated Primal-Dual algorithm

BFGS optimization

$$\theta^{(t+1)} = \theta^{(t)} - B^{(t)} \text{SUGAR}_{\theta^{(t)}}^{(k),\text{FDMC}}(y)$$







w_{true}

y

$\hat{w}_{\theta^\star}$

Summary

Estimator

$\hat{w}_\theta : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ estimator

$\theta \in \Theta$ hyper-parameter

$\mathcal{R} : \Theta \longrightarrow \mathbb{R}$ criterion

Goal

$$\text{Find } \theta^\star \in \operatorname*{argmin}_{\theta \in \Theta} \mathcal{R}(\theta)$$

(or close to it)

Estimator and algorithm

$$w_\theta^{(k)}(y) \longrightarrow \hat{w}_\theta(y)$$

$$\mathrm{Jac}_{\hat{w}_\theta}(y) \longrightarrow \mathrm{Jac}_{w_\theta^{(k)}}(y)$$

Iterative estimators

$$\mathrm{T} : \mathbb{R}^p \times \mathbb{R}^n \longrightarrow \mathbb{R}^p$$

$$\begin{cases} w_\theta^{(k)}(y) \xrightarrow{k \to +\infty} \hat{w}_\theta(y) \\ w_\theta^{(k+1)}(y) = \mathrm{T}(w_\theta^{(k)}(y), y) \end{cases}$$

Chain rule

$$\mathrm{Jac}_{w_\theta^{k+1}}(y) = \partial_1 \mathrm{T}(w_\theta^k(y), y) \cdot \mathrm{Jac}_{w_\theta^k(y)}(y) + \partial_2 \mathrm{T}(w_\theta^k(y), y)$$

"Hyper"-gradient descent

$$\theta^{k+1} = \theta^k - \rho \nabla \mathcal{R}(\theta^k)$$

Several strategies:

- Implicit differentiation
- Risk estimation (SUGAR)
- Direct Jacobian estimation