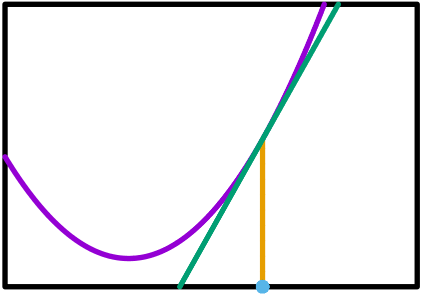
An Introduction to Online Convex Optimization



Wouter M. Koolen

CWI
Centrum Wiskunde & Informatica

ILPS Lunch, Friday 12$^{\text{th}}$ May, 2017

About me

Tenure tracker in CWI Machine Learning group on VENI grant

I work on Machine Learning Theory

- Online learning
- Easy data
- Game tree search

Adversarial Intelligence blog

`http://blog.wouterkoolen.info`

Local chair COLT'17 Amsterdam ← **check it out!**

Grand Goal of Machine Learning

Design systems that improve performance by learning from data.

$$\text{system} + \text{data} \; = \; \text{better system}$$

- Batch learning: training $\rightarrow$ production
- Online learning: continuously improving.

Overview

Overview of today's content

- Example
- OCO problem
- Classic algorithm for OCO
- Modern OCO developments

Example: spam classification (linear model)

- A new email arrives.
  Encoded as feature vector $\boldsymbol{x}_t$ (bag of words, ...)
- System assigns a spam rating $\boldsymbol{w}_t^\mathsf{T}\boldsymbol{x}_t$
  Puts it in **inbox** or spam folder
- User intervenes if misclassified ☹
  System gets actual label $y_t = \{-1, +1\}$

Example: spam classification (linear model)

- A new email arrives.
  Encoded as feature vector $\boldsymbol{x}_t$ (bag of words, ...)
- System assigns a spam rating $\boldsymbol{w}_t^\intercal \boldsymbol{x}_t$
  Puts it in **inbox** or spam folder
- User intervenes if misclassified ☹
  System gets actual label $y_t = \{-1, +1\}$

Question

How to pick $\boldsymbol{w}_t$?

Example: spam classification (linear model)

- A new email arrives.
  Encoded as feature vector $\boldsymbol{x}_t$ (bag of words, ...)
- System assigns a spam rating $\boldsymbol{w}_t^\intercal \boldsymbol{x}_t$
  Puts it in **inbox** or spam folder
- User intervenes if misclassified ☹
  System gets actual label $y_t = \{-1, +1\}$

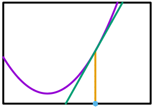
Question

How to pick $\boldsymbol{w}_t$?

We need a loss function. Variety of choices:

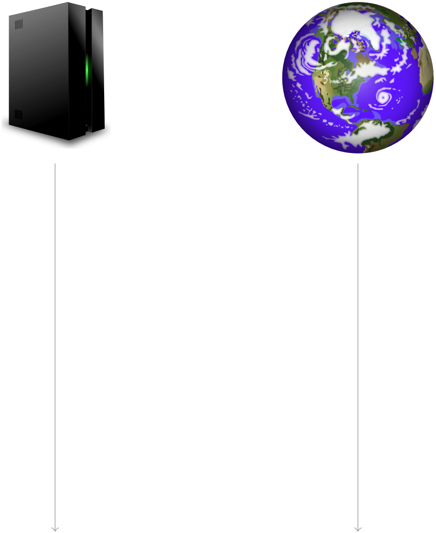
square loss	$(\boldsymbol{w}_t^\intercal \boldsymbol{x}_t - y_t)^2$
logistic loss	$-\ln\left(1 + e^{-y_t \boldsymbol{w}_t^\intercal \boldsymbol{x}_t}\right)$
hinge loss	$\max\{0, 1 - y_t \boldsymbol{w}_t^\intercal \boldsymbol{x}_t\}$

What is OCO and why is it useful?

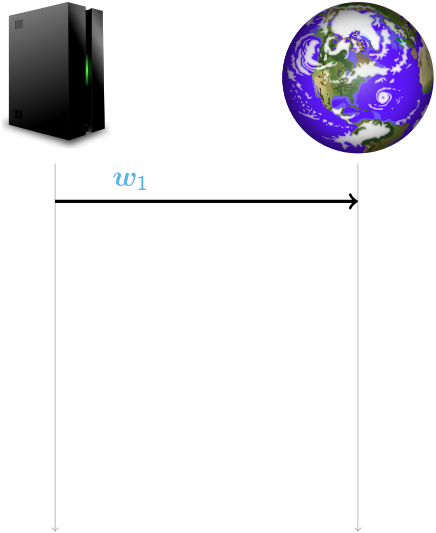


- Model for sequential decision making problems
  like spam filtering, portfolio investment, route planning, data compression, etc ...
- Close fit to a range of practical problems
- Very crisp (theoreticians)
- Many of the features of hard, complex problems
- Powerful and principled methods
- Basis of reductions
  - online to batch (for statistical learning)
  - bandits (for partial information problems)
  - saddle point problems (solving games)
  - non-convex problems

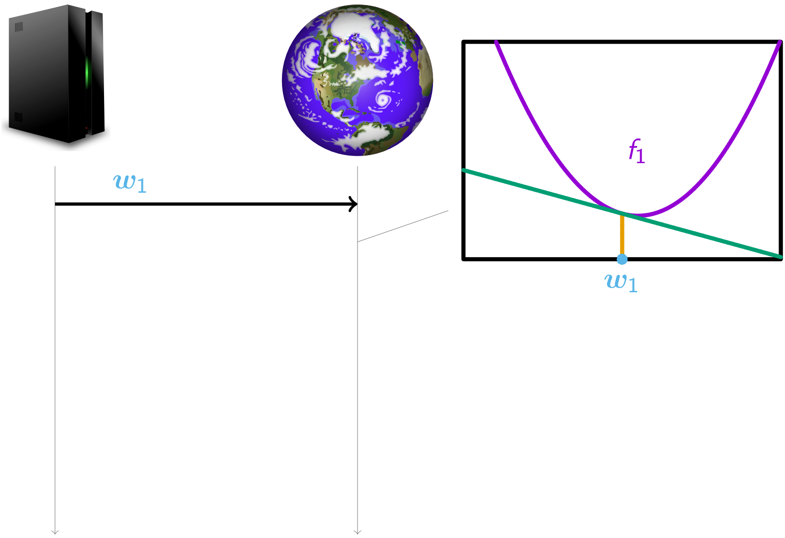
Online Convex Optimisation, Protocol



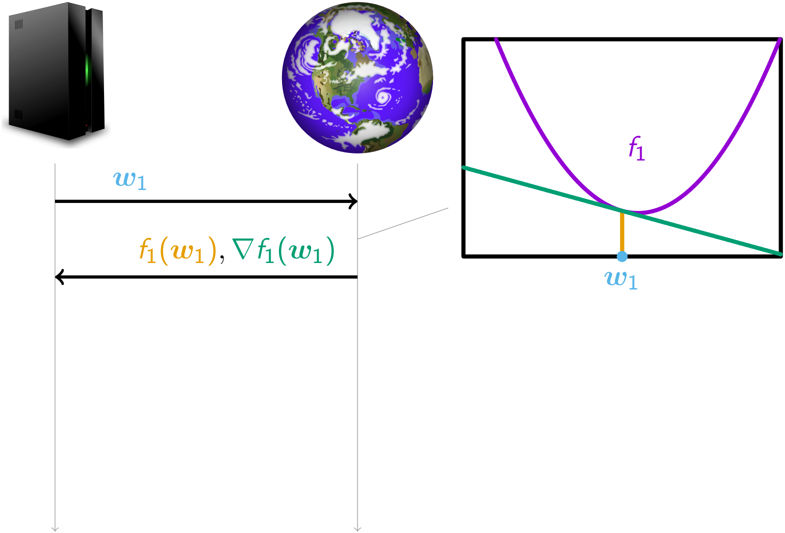
Online Convex Optimisation, Protocol



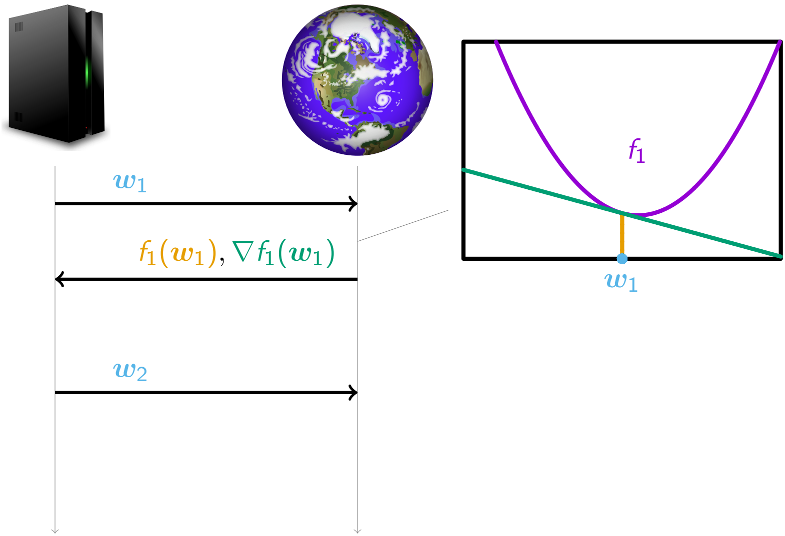
Online Convex Optimisation, Protocol



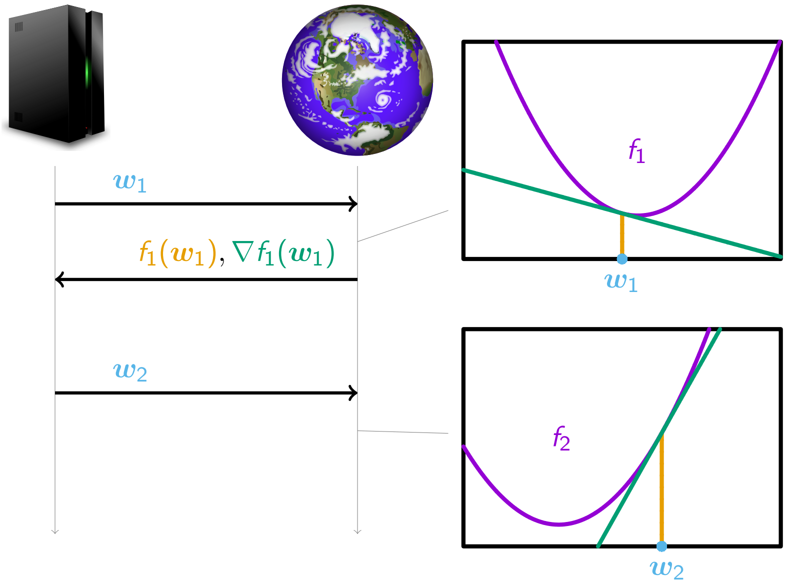
Online Convex Optimisation, Protocol



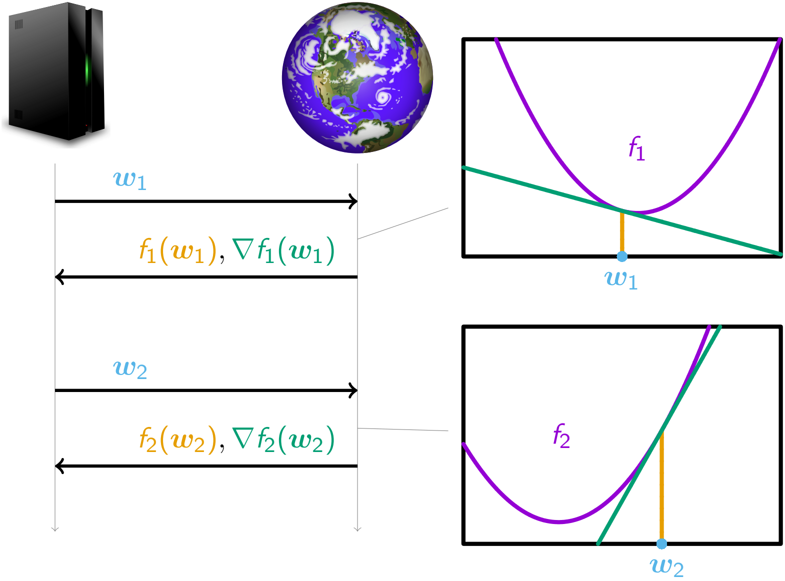
Online Convex Optimisation, Protocol



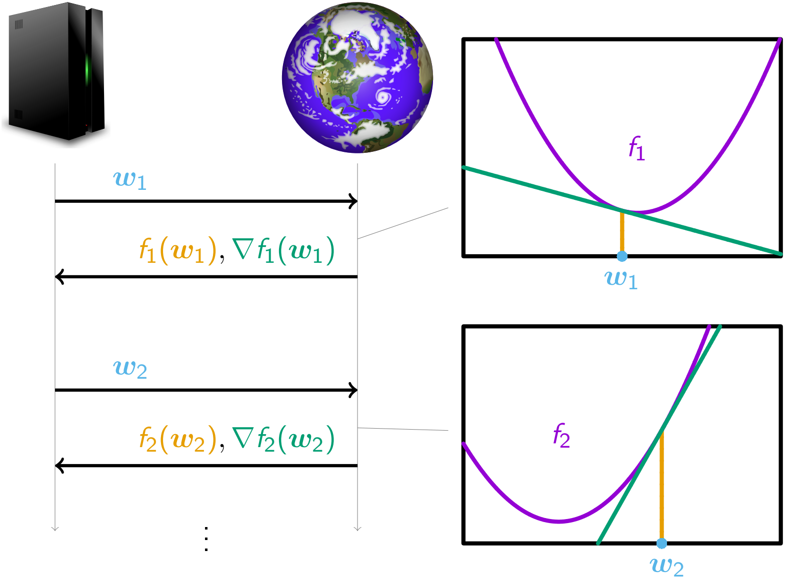
Online Convex Optimisation, Protocol



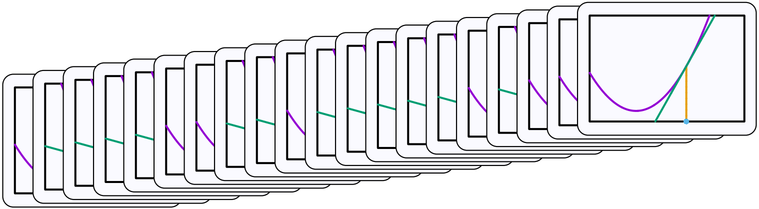
Online Convex Optimisation, Protocol



Online Convex Optimisation, Protocol



Online Convex Optimisation, Objective



Definition (Regret)

$$R_T = \underbrace{\sum_{t=1}^{T} f_t(\boldsymbol{w}_t)}_{\text{Online loss}} - \underbrace{\min_{\boldsymbol{u}} \sum_{t=1}^{T} f_t(\boldsymbol{u})}_{\text{Optimal loss}}$$

Online Gradient Descent [Zinkevich, 2003]

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla f_t(\boldsymbol{w}_t)$$

Online Gradient Descent [Zinkevich, 2003]

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla f_t(\boldsymbol{w}_t)$$

$$= \arg\min_{\boldsymbol{w}} \sum_{s=1}^{t} \boldsymbol{w}^\intercal \nabla f_s(\boldsymbol{w}_s) + \frac{1}{\eta} \|\boldsymbol{w}\|^2$$

Online Gradient Descent [Zinkevich, 2003]

$$
\begin{aligned}
\boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \eta \nabla f_t(\boldsymbol{w}_t) \\
&= \operatorname*{arg\,min}_{\boldsymbol{w}} \sum_{s=1}^{t} \boldsymbol{w}^\intercal \nabla f_s(\boldsymbol{w}_s) + \frac{1}{\eta} \|\boldsymbol{w}\|^2
\end{aligned}
$$

Worst-case regret guarantee:

$$
R_T = O\left(\sqrt{T}\right)
$$

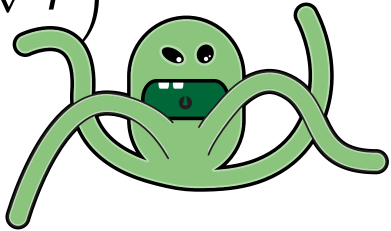
Online Gradient Descent [Zinkevich, 2003]

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla f_t(\boldsymbol{w}_t)$$

$$= \arg\min_{\boldsymbol{w}} \sum_{s=1}^{t} \boldsymbol{w}^\intercal \nabla f_s(\boldsymbol{w}_s) + \frac{1}{\eta} \|\boldsymbol{w}\|^2$$

Worst-case regret guarantee:

$$R_T = O\left(\sqrt{T}\right)$$



Modern OCO

Question

Can we improve performance by using more domain knowledge?

Modern OCO

Question

Can we improve performance by using more domain knowledge?

Refined measures of complexity of OCO problems

- Gradient norms (maybe the gradients vanish)
- Curvature (strongly convex, exp concave, mixable)
- Stochastic scenarios (not adversarial but friendly data)

Modern OCO

Question

Can we improve performance by using more domain knowledge?

Refined measures of complexity of OCO problems

- Gradient norms (maybe the gradients vanish)
- Curvature (strongly convex, exp concave, mixable)
- Stochastic scenarios (not adversarial but friendly data)

In both cases the key is

- Adaptive tuning of the learning rate
- Knowledge about the loss beyond convexity (add quadratic)

Go-to algorithms: AdaGrad, Online Newton Step, MetaGrad

Conclusion

I hope you got a flavour of OCO.

Happy to discuss in more detail.

Thanks!