WIC Wintermeeting, February 1, 2016

From Data Compression to Online Machine Learning

Tim van Erven

Based on joint work with:
Wouter Koolen, **Peter Grünwald**

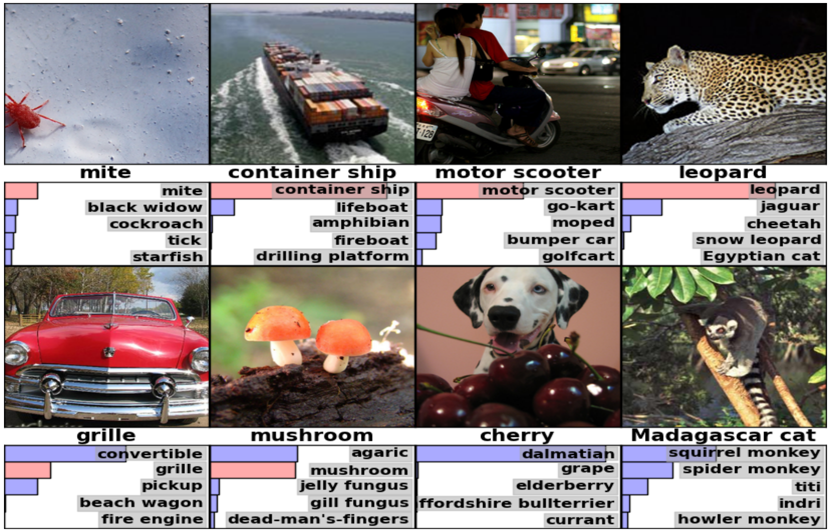
Universiteit Leiden

Outline

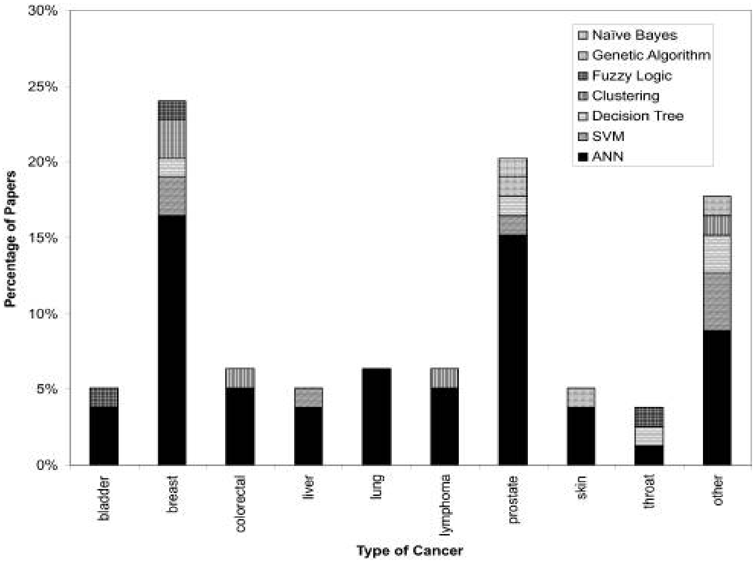
- **The end: online convex optimization for machine learning**
- The beginning: data compression and universal coding via sequential predictions
- Sequential predictions for general losses
- Online Convex Optimization

Machine Learning Examples

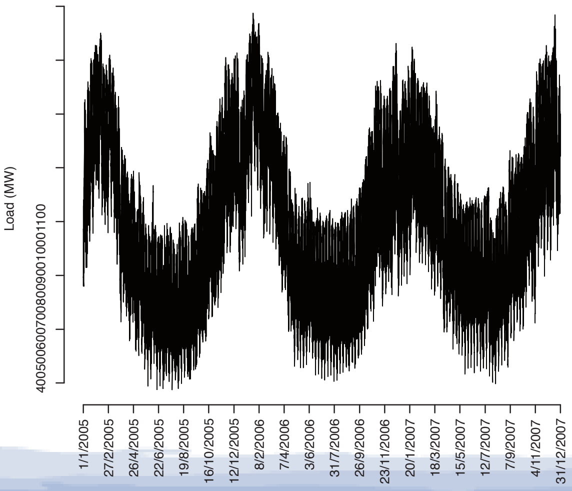
Image Classification



Cancer Research



Forecasting Electricity Consumption



E-mail Spam Detection

Machine Learning

- Training data: $\begin{pmatrix} Y_1 \\ \boldsymbol{X}_1 \end{pmatrix}, \ldots, \begin{pmatrix} Y_n \\ \boldsymbol{X}_n \end{pmatrix}$

desired response

input vector

- Many parameters: $\boldsymbol{v} = (v^1, \ldots, v^d)$

- Optimize performance on training data:

$$\min_{\boldsymbol{v}} \quad f_1(\boldsymbol{v}) + \ldots + f_n(\boldsymbol{v})$$

where f_t measures the loss/error on $\begin{pmatrix} Y_t \\ \boldsymbol{X}_t \end{pmatrix}$

e.g. logistic loss: $f_t(\boldsymbol{v}) = \log(1 + e^{-Y_t \langle \boldsymbol{v}, \boldsymbol{X}_t \rangle})$

Machine Learning

- Traini

Problems for **big data**:

- Data does not fit in **memory** at once
- Want to **update fast** on extra data

or

- Many

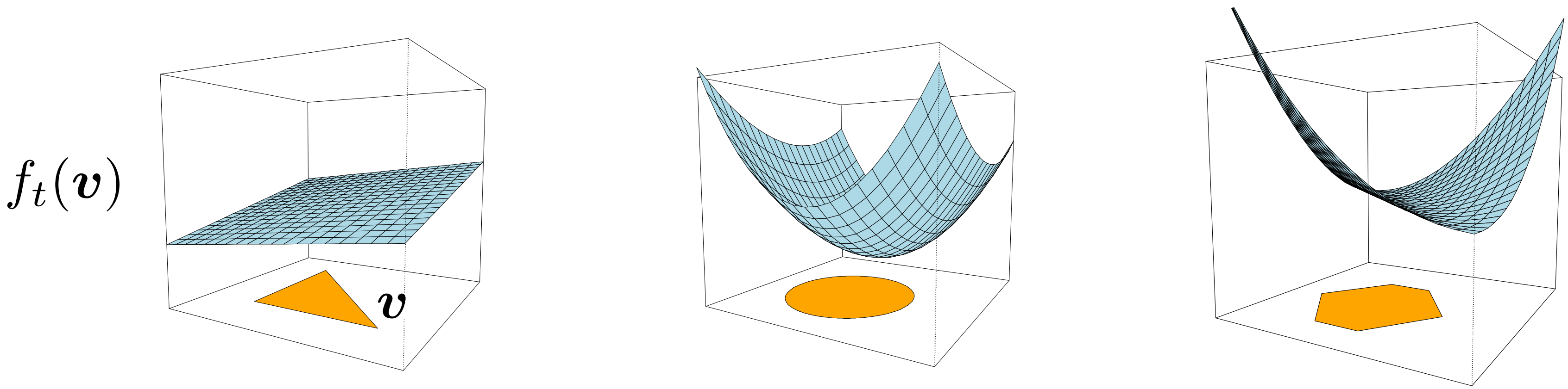
- Optimize performa on training data:

$$\min_{\boldsymbol{v}} \quad f_1(\boldsymbol{v}) + \ldots + f_n(\boldsymbol{v})$$

where f_t measures the loss/error on $\begin{pmatrix} Y_t \\ \boldsymbol{X}_t \end{pmatrix}$

e.g. logistic loss: $f_t(\boldsymbol{v}) = \log(1 + e^{-Y_t \langle \boldsymbol{v}, \boldsymbol{X}_t \rangle})$

Online Convex Optimization

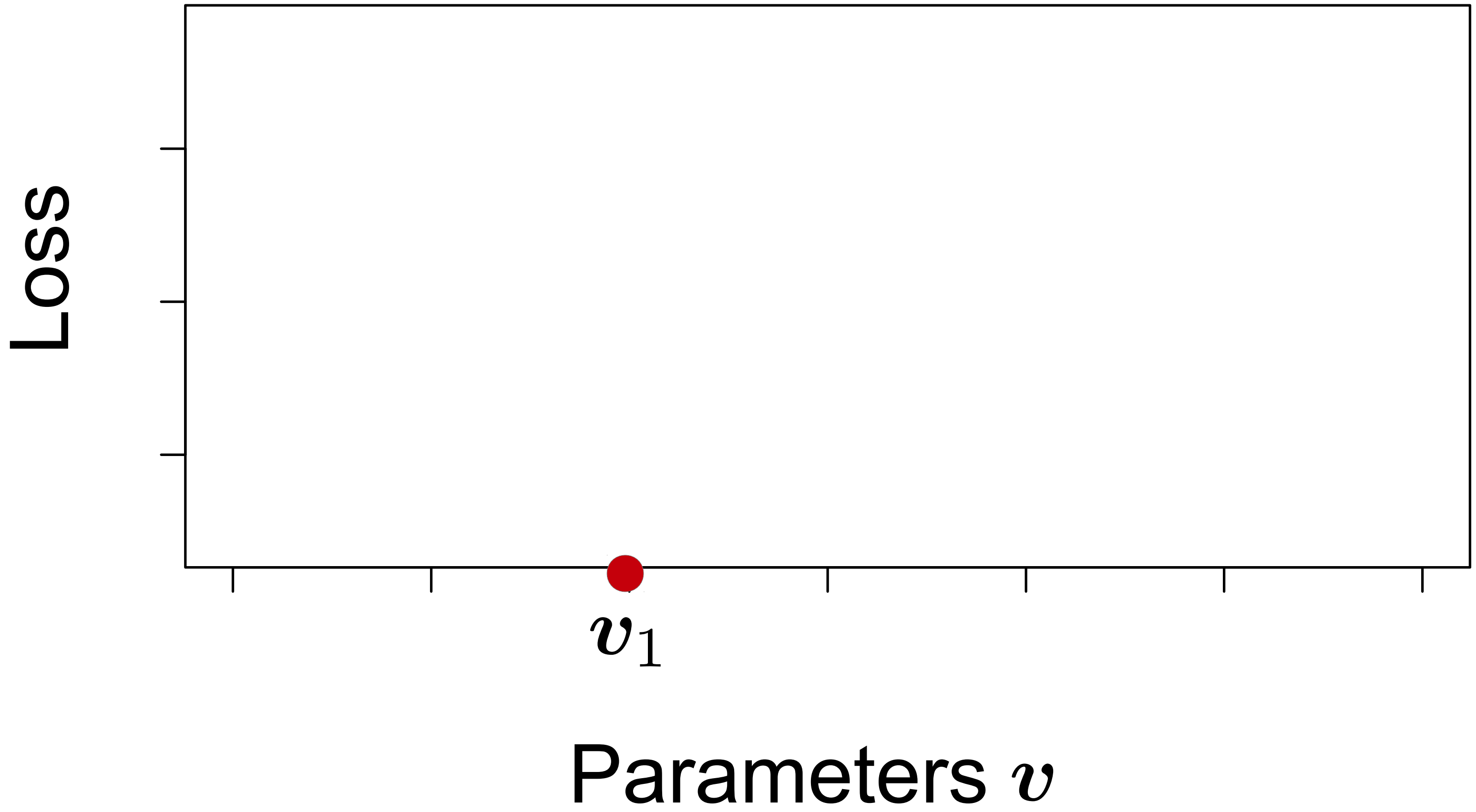


- **Convex** functions $f_1(\boldsymbol{v}), \ldots, f_n(\boldsymbol{v})$
- Process data sequentially:

  Continuously improve parameters $\boldsymbol{v}$
  by looking at **one function** f_t **at a time**

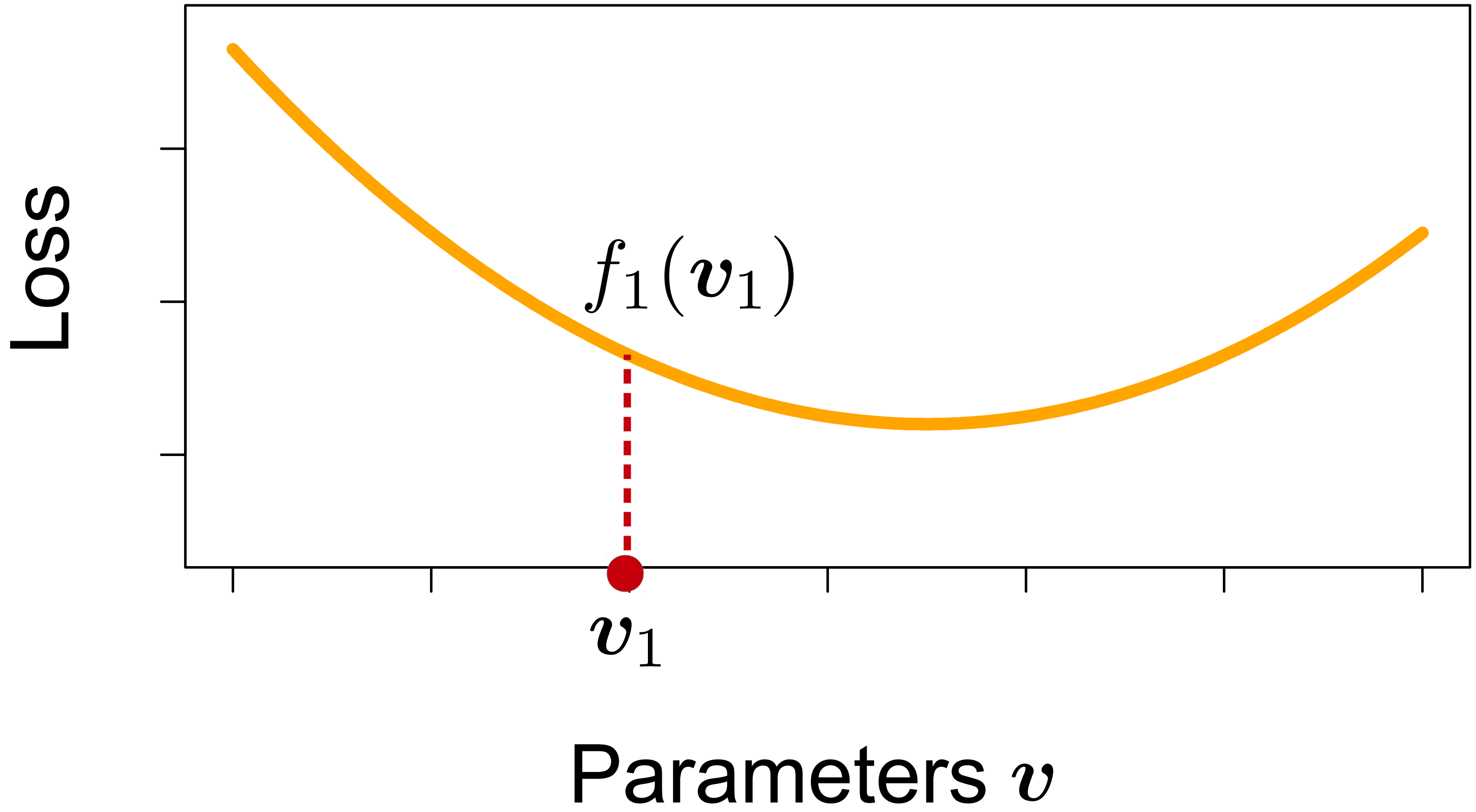
Online Gradient Descent

Initialize parameters



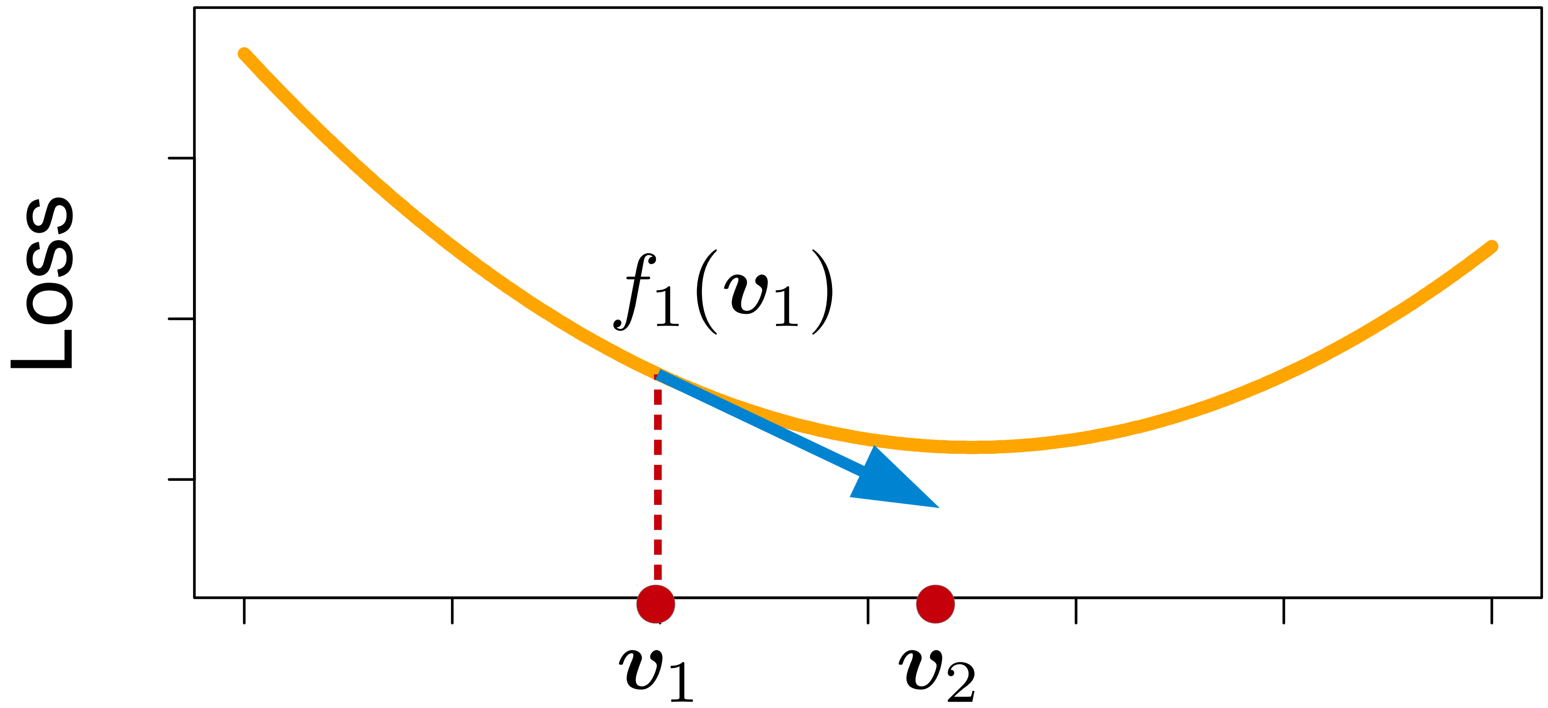
Online Gradient Descent

Round 1



Online Gradient Descent

Round 1

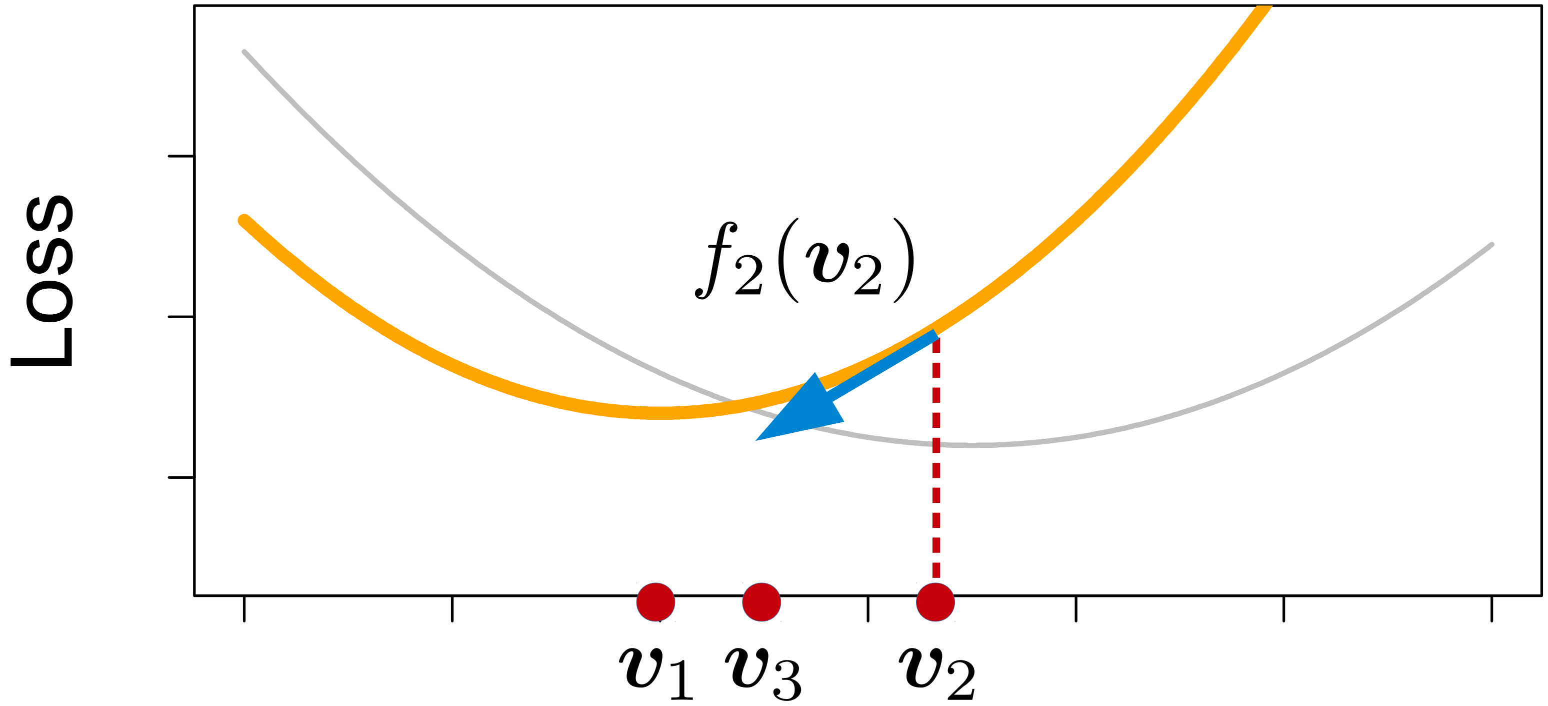


Move in **direction** of steepest descent

(step size controlled by parameter η)

Online Gradient Descent

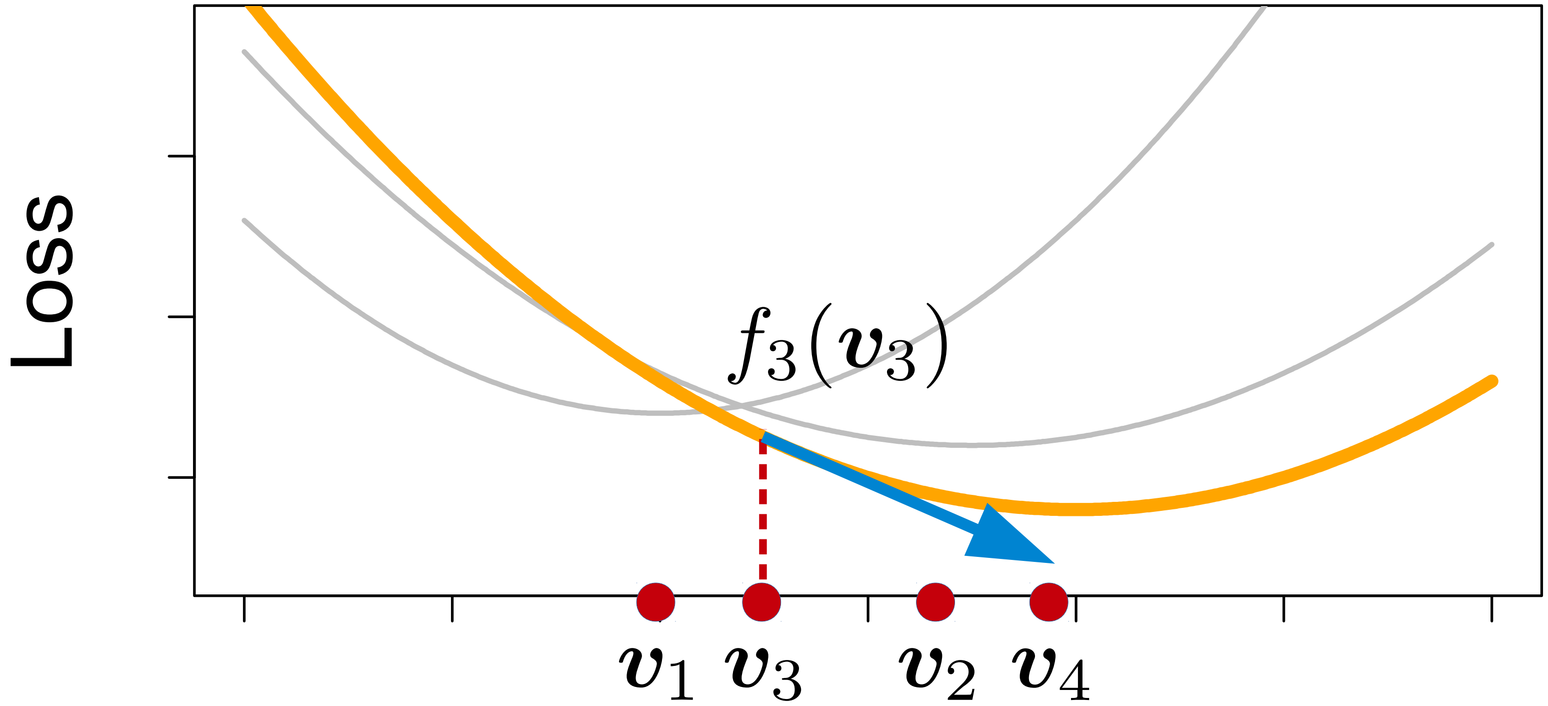
Round 2



Move in **direction** of steepest descent
(step size controlled by parameter η)

Online Gradient Descent

Round 3



Move in **direction** of steepest descent
(step size controlled by parameter η)

What does this have to do with **information theory**?

Outline

- The end: online convex optimization for machine learning
- **The beginning: data compression and universal coding via sequential predictions**
- Sequential predictions for general losses
- Online Convex Optimization

Data Compression via Sequential Prediction

- Data: $X_1, \ldots, X_n$
- Encode in sequential pass through the data
- For $t = 1, \ldots, n$:
  - Predict X_t by distribution $\hat{P}_t$
  - Encode X_t with $-\log \hat{P}_t(X_t)$ bits

- $\hat{P}_t$ depends only on previous data $X_1, \ldots, X_{t-1}$
- Efficient algorithm: arithmetic coding

Universal Coding

- Suppose we have K prediction strategies/codes $P_t^1, \ldots, P_t^K$
- How to predict/code (nearly) as well as the best one?

- **Regret** = our codelength – codelength of best

$$= \sum_{t=1}^{n} -\log \hat{P}_t(X_t) - \min_k \sum_{t=1}^{n} -\log P_t^k(X_t)$$

Bayesian Predictions for Universal Coding

- Start with uniform **prior distribution** $w_1(k) = \frac{1}{K}$ on K prediction strategies
- Predict with Bayes predictive distribution, which **mixes** strategies

$$\hat{P}_t(X_t) = \Pr(X_t | X_1, \ldots, X_{t-1}) = \sum_{k=1}^{K} w_t(k) P_t^k(X_t)$$

according to **posterior probabilities**

$$w_t(k) = \frac{w_1(k) \prod_{s=1}^{t-1} P_s^k(X_s)}{\text{normalization}}$$

Regret Bound for Bayesian Predictions

- **Regret** = our codelength – codelength of best

  $= \sum_{t=1}^{n} -\log \hat{P}_t(X_t) - \min_k \sum_{t=1}^{n} -\log P_t^k(X_t)$

  $\leq \log K$

- **Proof**: let k^* be the best strategy. Then our predictions satisfy

$$\prod_{t=1}^{n} \hat{P}_t(X_t) = \prod_{t=1}^{n} \Pr(X_t | X_1, \ldots, X_{t-1}) = \Pr(X_{1:n})$$

$$= \sum_{k=1}^{k} w_1(k) \Pr(X_{1:n}|k) \geq w_1(k^*) \Pr(X_{1:n}|k^*) = \frac{1}{K} \prod_{t=1}^{n} P_t^{k^*}(X_t)$$

Outline

- The end: online convex optimization for machine learning
- The beginning: data compression and universal coding
  via sequential predictions
- **Sequential predictions for general losses:**
  - Log loss = data compression
  - Exp-concave losses
  - Linear loss
- Online Convex Optimization

Sequential Prediction for General Losses

- Suppose we have K prediction strategies that make predictions $p_t^1, \ldots, p_t^K$ in round t
- Do **not** have to be probabilities

- For $t = 1, \ldots, n$:
  - Predict $\hat{p}_t$
  - $\mathrm{loss}_t(p)$ measures loss of p on outcome X_t

- Regret = $\sum_{t=1}^{n} \mathrm{loss}_t(\hat{p}_t) - \min_k \sum_{t=1}^{n} \mathrm{loss}_t(p_t^k)$

Sequential Prediction for General Losses

Data compression:
- Predictions are prob. distributions
- $\text{loss}_t(p) = -\log p(X_t)$ is **log loss**

Regression:
- Predictions are numbers
- $\text{loss}_t(p) = (X_t - p)^2$ is **squared error**

- Suppo … at make …
- Do **no** …
- For $t = 1, \ldots$
  - Predict $\hat{p}_t$
  - $\text{loss}_t(p)$ measures loss of p on outcome X_t
- Regret = $\sum_{t=1}^{n} \text{loss}_t(\hat{p}_t) - \min_k \sum_{t=1}^{n} \text{loss}_t(p_t^k)$

Outline

- The end: online convex optimization for machine learning
- The beginning: data compression and universal coding via sequential predictions
- Sequential predictions for general losses:
  - Log loss = data compression
  - **Exp-concave losses**
  - Linear loss
- Online Convex Optimization

Exp-concave Losses

- Losses such than

$$e^{-\eta \, \mathrm{loss}_t(p)}$$

is **concave** in our prediction p for some $\eta > 0$

- **Log loss**: $e^{-\mathrm{loss}_t(p)} = p(X_t)$
  - linear in p for $\eta = 1$
- **Squared error**: $e^{-\eta(X_t - p)^2}$
  - $\eta = \frac{1}{8}$ if $X_t, p \in [-1, +1]$

Exp-concave Losses

- Losses such that

Behaves much like a probability

$$e^{-\eta \operatorname{loss}_t(p)}$$

is **concave** in our prediction p for some $\eta > 0$

- **Log loss**: $e^{-\operatorname{loss}_t(p)} = p(X_t)$
  - linear in p for $\eta = 1$
- **Squared error**: $e^{-\eta(X_t - p)^2}$
  - $\eta = \frac{1}{8}$ if $X_t, p \in [-1, +1]$

Exp-concavity allows mixing “probabilities”

- If we mix predictions according to some weights:

$$\hat{p}_t = \sum_{k=1}^{K} w_t(k) p_t^k$$

- Then our “probability” is at least the mixture of the “probabilities” we are mixing:

$$e^{-\eta \, \mathrm{loss}_t(\hat{p}_t)} \geq \sum_{k=1}^{K} w_t(k) e^{-\eta \, \mathrm{loss}_t(p_t^k)}$$

Exponential Weights Predictions

- Predict with Bayesian predictions, which **mix** strategies

$$\hat{p}_t = \sum_{k=1}^{K} w_t(k) p_t^k$$

according to **posterior weights**

$$w_t(k) = \frac{w_1(k) \prod_{s=1}^{t-1} \boldsymbol{p_s^k(X_s)}}{\text{normalization}}$$

Exponential Weights Predictions

- Predict with Bayesian predictions, which **mix** strategies

$$\hat{p}_t = \sum_{k=1}^{K} w_t(k) p_t^k$$

according to **posterior weights**

$$w_t(k) = \frac{w_1(k) \prod_{s=1}^{t-1} e^{-\eta \,\mathrm{loss}_s (p_s^k)}}{\mathrm{normalization}}$$

Regret for Exp-Concave Losses

- **Regret** = our total loss – loss of best strategy

$$= \sum_{t=1}^{n} \mathrm{loss}_t(\hat{p}_t) - \min_k \sum_{t=1}^{n} \mathrm{loss}_t(p_t^k)$$

$$\leq \frac{\log K}{\eta}$$

- **Proof**: same steps as for log loss give

$$\sum_{t=1}^{n} \eta \, \mathrm{loss}_t(\hat{p}_t) \leq \sum_{t=1}^{n} \eta \, \mathrm{loss}_t(p_t^{k^*}) + \log K$$

Outline

- The end: online convex optimization for machine learning
- The beginning: data compression and universal coding via sequential predictions
- Sequential predictions for general losses:
  - Log loss = data compression
  - Exp-concave losses
  - **Linear loss**
- Online Convex Optimization

Linear Loss

- Predict with a **mix** of K prediction strategies:

$$\hat{p}_t = \sum_{k=1}^{K} w_t(k) p_t^k$$

- Loss is **linear in the mixing weights**:

$$\mathrm{loss}_t(\boldsymbol{w}_t) = \sum_{k=1}^{K} w_t(k) \ell_t^k$$

where ℓ_t^k is the loss of using strategy k
(can be anything)

- Example: strategies classify emails as spam or not spam

$$\ell_t^k = \begin{cases} 1 & \text{if strategy k makes mistake on t-th e-mail,} \\ 0 & \text{otherwise} \end{cases}$$

Regret for Linear Loss

- Can **approximate** linear loss by an **exp-concave loss** $m_t(\boldsymbol{w})$ with parameter η
- **Approximation error**: $\eta/8$ per round (if $\ell_t^k \in [0,1]$)

- Exponential weights algorithm with $\eta = \sqrt{\frac{8\log(K)}{n}}$ achieves

$$\text{Regret} \leq \frac{\log K}{\eta} + \frac{n\eta}{8} = \sqrt{n\log(K)/2}$$

$m_t(\boldsymbol{w}) = -\frac{1}{\eta}\log\sum_k w(k)e^{-\eta\ell_t^k}$

Outline

- The end: online convex optimization for machine learning
- The beginning: data compression and universal coding via sequential predictions
- Sequential predictions for general losses
- **Online Convex Optimization**
  - Linear optimization
  - Convex optimization

Online Linear Optimization

- Linear loss with an **infinite** number of **comparison strategies** $\boldsymbol{v} \in \mathbb{R}^d$
- Loss of $\boldsymbol{v}$ in round t is

$$\ell_t^{\boldsymbol{v}} = \langle \boldsymbol{v}, \boldsymbol{c}_t \rangle \qquad \text{for some costs } \boldsymbol{c}_t \in \mathbb{R}^d$$

- Our loss with weights $w_t(\boldsymbol{v})$ is

$$\text{loss}_t(w_t) = \langle \boldsymbol{\mu}_t, \boldsymbol{c}_t \rangle$$

where $\boldsymbol{\mu}_t = \mathbb{E}_{w_t(\boldsymbol{v})}[\boldsymbol{v}]$ is the mean of w_t

Exponential Weights

- Exponential weights with **Gaussian prior**

$$w_1 = \mathcal{N}(0, I)$$

gives **Gaussian posterior weights**

$$w_t(\boldsymbol{v}) = \frac{w_1(\boldsymbol{v}) \prod_{s=1}^{t-1} e^{-\eta \langle \boldsymbol{v}, \boldsymbol{c}_s \rangle}}{\text{normalization}} = \mathcal{N}(\boldsymbol{\mu}_t, I)$$

with **mean**

$$\boldsymbol{\mu}_t = -\eta \sum_{s=1}^{t-1} \boldsymbol{c}_s$$

Regret for Linear Optimization

- Thm: If $\|\boldsymbol{c}_t\| \leq 1$ for all t. Then the regret of **exponential weights** with

$$\eta = \sqrt{\frac{B^2}{n}}$$

with respect to all $\boldsymbol{v}$ s.t. $\|\boldsymbol{v}\| \leq B$ is at most

$$\text{Regret} \leq \sqrt{2B^2 n}$$

- Essentially **same analysis** as for finite number of comparison strategies

Outline

- The end: online convex optimization for machine learning
- The beginning: data compression and universal coding via sequential predictions
- Sequential predictions for general losses
- Online Convex Optimization
  - Linear optimization
  - **Convex optimization**

Machine Learning

- Training data: $\begin{pmatrix} Y_1 \\ \boldsymbol{X}_1 \end{pmatrix}, \ldots, \begin{pmatrix} Y_n \\ \boldsymbol{X}_n \end{pmatrix}$

desired response

input vector

- Many parameters: $\boldsymbol{v} = (v^1, \ldots, v^d)$

- Optimize performance on training data:

$$\min_{\boldsymbol{v}} \quad f_1(\boldsymbol{v}) + \ldots + f_n(\boldsymbol{v})$$

where f_t measures the loss/error on $\begin{pmatrix} Y_t \\ \boldsymbol{X}_t \end{pmatrix}$

e.g. logistic loss: $f_t(\boldsymbol{v}) = \log(1 + e^{-Y_t \langle \boldsymbol{v}, \boldsymbol{X}_t \rangle})$

Online Convex Optimization

- Loss of $\boldsymbol{v} \in \mathbb{R}^d$ in round t is

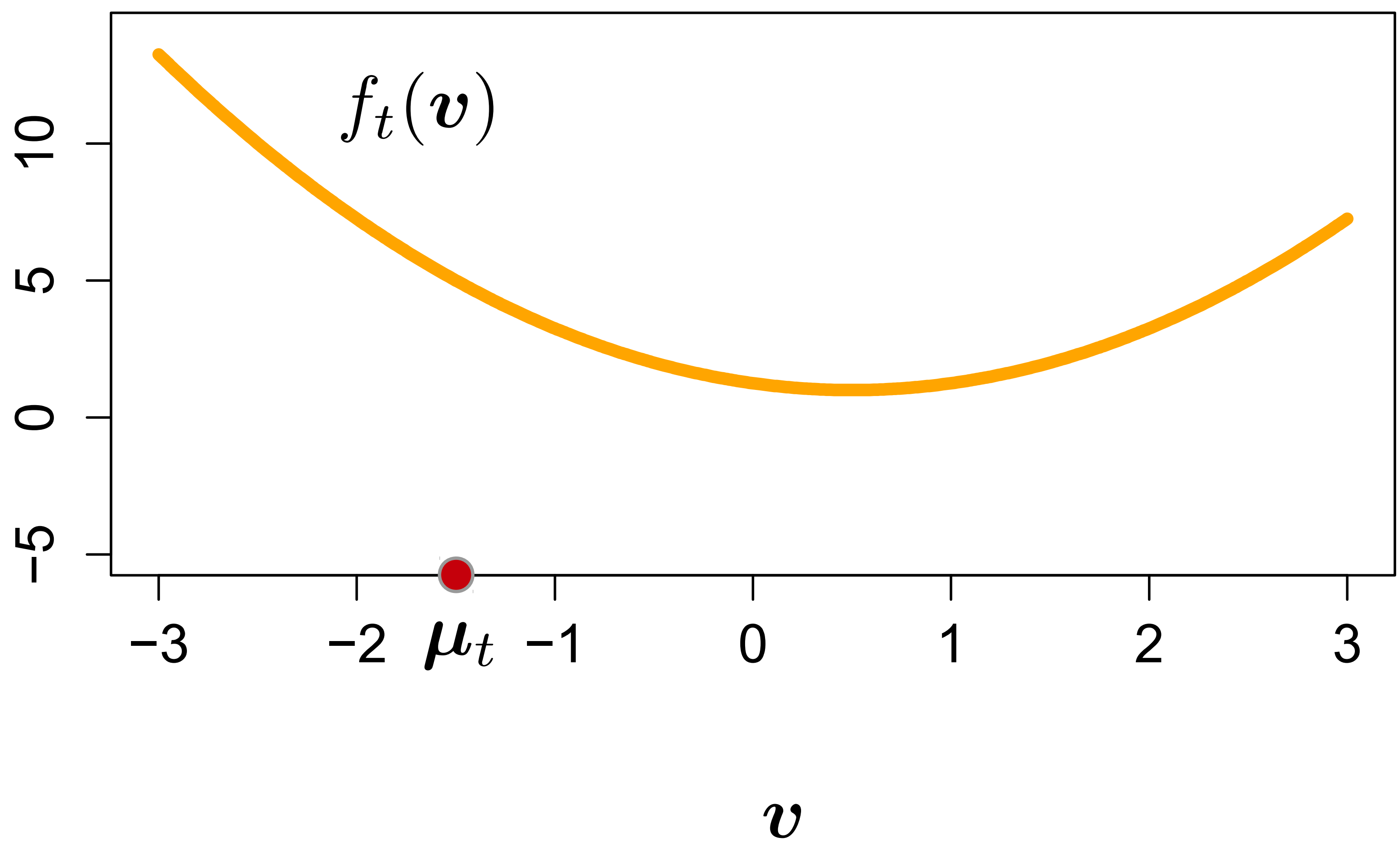
$$\ell_t^{\boldsymbol{v}} = f_t(\boldsymbol{v}) \qquad \text{for convex } f_t$$

- Our loss with weights $w_t(\boldsymbol{v})$ is

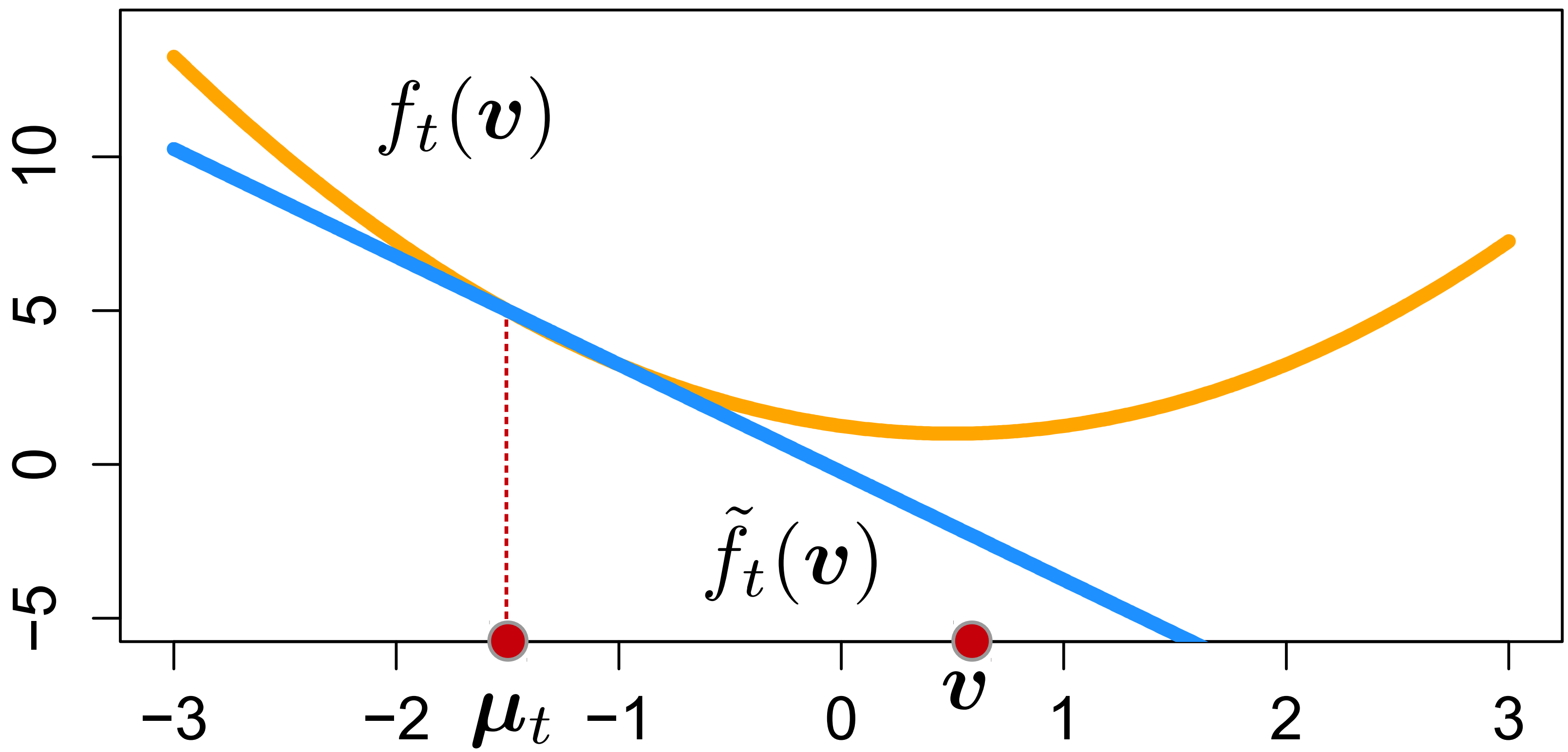
$$\mathrm{loss}_t(w_t) = f_t(\boldsymbol{\mu}_t)$$

- Regret = $\sum_{t=1}^{n} f_t(\boldsymbol{\mu}_t) - \min_{\boldsymbol{v}} \sum_{t=1}^{n} f_t(\boldsymbol{v})$

Reduction to Linear Optimization



Reduction to Linear Optimization



Approximate convex orange by linear blue

$$\tilde{f}_t(\boldsymbol{v}) = f_t(\boldsymbol{\mu}_t) + \langle (\boldsymbol{v} - \boldsymbol{\mu}_t), \nabla f_t(\boldsymbol{\mu}_t) \rangle$$

Exponential Weights becomes Gradient Descent

- Effect of **linear approximation**:

$$\boldsymbol{c}_t = \nabla f_t(\boldsymbol{\mu}_t)$$

- Mean of exponential weights becomes

$$\boldsymbol{\mu}_t = -\eta \sum_{s=1}^{t-1} \nabla f_s(\boldsymbol{\mu}_s) = \boldsymbol{\mu}_{t-1} - \eta \nabla f_{t-1}(\boldsymbol{\mu}_{t-1})$$

which is exactly **gradient descent**!

Regret for Convex Optimization

- Thm: If $\|\nabla f_t(\boldsymbol{\mu}_t)\| \leq 1$ for all t. Then the regret of exponential weights = gradient descent with

$$\eta = \sqrt{\frac{B^2}{n}}$$

with respect to all $\boldsymbol{v}$ s.t. $\|\boldsymbol{v}\| \leq B$ is at most

$$\sum_{t=1}^{n} f_t(\boldsymbol{\mu}_t) - \min_{\boldsymbol{v}:\ \|\boldsymbol{v}\| \leq B} \sum_{t=1}^{n} f_t(\boldsymbol{v}) \leq \sqrt{2B^2 n}$$

Summary

- **Generalize universal coding** to:
  - sequential prediction with general losses
  - online convex optimization
    (for machine learning)

- **Same algorithm** everywhere:
  - Bayesian posterior weights (universal coding)
  - Exponential weights
  - Online gradient descent

Recent Developments

Joint work with **Wouter Koolen**

- Exponential weights/gradient descent:
  - Tune parameter η to optimize **bound**
- New algorithm 'Squint':
  - **Improved exponential weights** for sequential prediction with linear losses
  - Automatically **learns optimal parameter** η for the data
  - Replaces $\sqrt{n}$ by **variance measure** $\sqrt{V} \ll \sqrt{n}$
- Work in progress: transfer results to the online convex optimization setting

References

- **Standard textbook on online learning**:
  Cesa-Bianchi and Lugosi. Prediction, learning, and games. 2006.
- **Online convex optimization tutorial**:
  Shalev-Shwartz. Online Learning and Online Convex Optimization. Foundations and Trends in Machine Learning. Vol. 4, No. 2 pp 107–194, 2011.
- **Recent developments**:
  Koolen, Van Erven. Second-order Quantile Methods for Experts and Combinatorial Games. Proceedings of the 28th Conference on Learning Theory (COLT), pp. 1155-1175, 2015.