Combining Adversarial Guarantees and Fast Rates in Online Learning

http://bitbucket.org/wmkoolen/metagrad

Wouter M. Koolen

Joint work with Tim van Erven and Peter Grünwald

Université Toulouse III - Paul Sabatier
Thursday 6th April, 2017
In a Nutshell

MetaGrad
optimisation alg.

Worst case

Stochastic data

Curvature

...
1. The online convex optimisation problem
2. State of the art
   - A taxonomy of losses
   - What’s missing?
3. Main result: MetaGrad
   - Second order bound
   - Efficient implementation
4. Applications
   - Curvature
   - Stochastic case
   - Experiments
Optimisation Pervasive in Machine Learning

\[
\min_w \sum_{t=1}^{T} f_t(w)
\]
Optimisation Pervasive in Machine Learning

\[
\min_w \sum_{t=1}^{T} f_t(w)
\]

Batch Training (classification)
Optimisation Pervasive in Machine Learning

\[ \min_{\mathbf{w}} \sum_{t=1}^{T} f_t(\mathbf{w}) \]

Batch Training (classification)  
Time Series (investment)
Optimisation Pervasive in Machine Learning

\[ \min_w \sum_{t=1}^{T} f_t(w) \]

- Batch Training (classification)
- Time Series (investment)
- Big Data
Online Convex Optimisation

$w_1 f_1(w_1), \nabla f_1(w_1)$

$w_2 f_2(w_2), \nabla f_2(w_2)$
Online Convex Optimisation

\[ w_1 f_1(w_1), \nabla f_1(w_1) \]

\[ w_2 f_2(w_2), \nabla f_2(w_2) \]

\[ f_1 w_1 f_2 w_2 \ldots \]
Online Convex Optimisation
Online Convex Optimisation

\[ w_1, \nabla f_1(w_1), f_1, w_2, f_2(w_2), \nabla f_2(w_2) \ldots \]
Online Convex Optimisation

\[ f_1(w_1), \nabla f_1(w_1) \]

\[ w_1 \]

\[ w_2 \]

\[ f_1 \]

\[ w_1 \]
Online Convex Optimisation

\[ f_1 (w_1), \nabla f_1 (w_1) \]

\[ f_2 (w_2), \nabla f_2 (w_2) \]
Online Convex Optimisation

\[ f_1(w_1), \nabla f_1(w_1) \]

\[ f_2(w_2), \nabla f_2(w_2) \]
Online Convex Optimisation

\[ f_1(w_1), \nabla f_1(w_1) \]

\[ f_2(w_2), \nabla f_2(w_2) \]

\[ w_1 \]

\[ w_2 \]
Objective

Definition (Regret)

\[ R_T = \sum_{t=1}^{T} f_t(w_t) - \min_u \sum_{t=1}^{T} f_t(u) \]

- Online loss
- Optimal loss
Online Gradient Descent [Zinkevich, 2003]

\[ \mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t) \]
Online Gradient Descent [Zinkevich, 2003]

\[ \mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t) \]

Worst-case regret guarantee:

\[ R_T = O\left(\sqrt{T}\right) \]
Online Gradient Descent [Zinkevich, 2003]

\[ \mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t) \]

Worst-case regret guarantee:

\[ R_T = O \left( \sqrt{T} \right) \]
Loss Taxonomy $\sim$ Curvature

- **Convex**
  - linear, hinge, absolute

- **Exp-concave**
  - logistic, squared

- **Strongly convex**
  - squared distance

Worst-case regret

- $\sqrt{T}$
- $\frac{d \ln T}{(w \in \mathbb{R}^d)}$
- $\ln T$
Loss Taxonomy ~ Curvature

Convex
linear, hinge, absolute

Exp-concave
logistic, squared

Strongly convex
squared distance

Online Gradient Descent
[Zinkevich, 2003]

Online Gradient Descent
[Hazan et al., 2007]

Online Newton Step
[Hazan et al., 2007]
**Loss Taxonomy ~ Curvature**

- **Convex**
  - linear, hinge, absolute

- **Exp-concave**
  - logistic, squared
  
- **Strongly convex**
  - squared distance

Worst-case regret:
- \( \sqrt{T} \)
- \( d \ln T \) \((w \in \mathbb{R}^d)\)
- \( \ln T \)

- **Online Gradient Descent** [Zinkevich, 2003]
- **Online Newton Step** [Hazan et al., 2007]
Loss Taxonomy $\sim$ Curvature

Worst-case regret

Convex
linear, hinge, absolute

Exp-concave
logistic, squared

Strongly convex
squared distance

Online Gradient Descent [Zinkevich, 2003]

Online Newton Step [Hazan et al., 2007]

Online Gradient Descent [Hazan et al., 2007]

$d \ln T$ ($w \in \mathbb{R}^d$)
Big Questions

Can we make **adaptive** methods for **online convex optimisation** that are

- **worst-case safe**
- exploit **curvature** automatically
- computationally **efficient**
Big Questions

Can we make adaptive methods for online convex optimisation that are

- worst-case safe
- exploit curvature automatically
- computationally efficient

And can we adapt to other important regimes?

- Mixed or in-between cases?
- Stochastic data? Bandits [Seldin and Slivkins, 2014]
- Absence of curvature? Experts [Koolen and Van Erven, 2015]
Main Idea

For every optimisation algorithm tuning is crucial.

Key obstacle: avoid learning \( \eta \) at slow rate.

Breakthrough: Multiple Eta Gradient algorithm (MetaGrad)
Main Idea

For every optimisation algorithm tuning is **crucial**.

So let’s **learn** optimal tuning from **data**.
Main Idea

For every optimisation algorithm tuning is crucial.

So let’s learn optimal tuning from data.

Key obstacle: avoid learning $\eta$ at slow rate itself.
Main Idea

For every optimisation algorithm tuning is **crucial**.

So let’s **learn** optimal tuning from **data**.

Key obstacle: avoid learning $\eta$ at **slow rate** itself.

Breakthrough: **Multiple Eta Gradient** algorithm (MetaGrad)
The regret of MetaGrad is bounded by

\[ R_T = O \left( \min \left\{ \sqrt{T}, \sqrt{V_T d \ln T} \right\} \right), \]

where

\[ V_T = \sum_{t=1}^{T} \left( (w_t - u^*)^T \nabla f_t(w_t) \right)^2 \]

measures variance compared to the offline optimum

\[ u^* = \arg \min_u \sum_{t=1}^{T} f_t(u) \]

Note: Optimal tuning depends on unknown optimum \( u^* \).
Proof Ideas

Analysis based on second-order surrogate loss. For each $\eta$:

$$\ell_t^\eta(u) := \eta(u - w_t)^T g_t + \eta^2 ((u - w_t)^T g_t)^2$$
Proof Ideas

Analysis based on second-order surrogate loss. For each \( \eta \):

\[
\ell^n_t(u) := \eta(u - w_t)^T g_t + \eta^2((u - w_t)^T g_t)^2
\]

Since surrogate is exp-concave for each fixed \( \eta \), we can use online quasi-Newton method like Online Newton Step [Hazan et al., 2007] to get predictions \( w^n_t \) that achieve logarithmic regret:

\[
\sum_{t=1}^{T} \ell^n_t(w^n_t) - \sum_{t=1}^{T} \ell^n_t(u) \leq O(d \ln T) \quad \forall u \in U
\]
Proof Ideas

Analysis based on second-order surrogate loss. For each $\eta$:

$$\ell_t^\eta(u) := \eta(u - w_t)^T g_t + \eta^2 ((u - w_t)^T g_t)^2$$

Since surrogate is exp-concave for each fixed $\eta$, we can use online quasi-Newton method like Online Newton Step [Hazan et al., 2007] to get predictions $w_t^\eta$ that achieve logarithmic regret:

$$\sum_{t=1}^{T} \ell_t^\eta(w_t^\eta) - \sum_{t=1}^{T} \ell_t^\eta(u) \leq O(d \ln T) \quad \forall u \in U$$

To learn the best $\eta$ we combine the predictions $w_t^\eta$ for multiple $\eta$ into a single master prediction $w_t$ using an experts algorithm for combining multiple learning rates similar to Squint [Koolen and Van Erven, 2015], to get:

$$\sum_{t=1}^{T} \ell_t^\eta(w_t) - \sum_{t=1}^{T} \ell_t^\eta(w_t^\eta) \leq O(\ln \ln T) \quad \forall \eta$$

Difficulty: Master has to perform well under multiple loss functions simultaneously. No standard experts algorithm works!
Proof Ideas

Analysis based on second-order surrogate loss. For each $\eta$:

$$\ell_t^\eta(u) := \eta(u - w_t)^T g_t + \eta^2 ((u - w_t)^T g_t)^2$$

Since surrogate is exp-concave for each fixed $\eta$, we can use online quasi-Newton method like Online Newton Step [Hazan et al., 2007] to get predictions $w_t^\eta$ that achieve logarithmic regret:

$$\sum_{t=1}^{T} \ell_t^\eta(w_t^\eta) - \sum_{t=1}^{T} \ell_t^\eta(u) \leq O(d \ln T) \quad \forall u \in \mathcal{U}$$

To learn the best $\eta$ we combine the predictions $w_t^\eta$ for multiple $\eta$ into a single master prediction $w_t$ using an experts algorithm for combining multiple learning rates similar to Squint [Koolen and Van Erven, 2015], to get:

$$\sum_{t=1}^{T} \ell_t^\eta(w_t) - \sum_{t=1}^{T} \ell_t^\eta(w_t^\eta) \leq O(\ln \ln T) \quad \forall \eta$$

Difficulty: Master has to perform well under multiple loss functions simultaneously. No standard experts algorithm works!
Proof Ideas

Analysis based on second-order surrogate loss. For each $\eta$:

$$\ell_t^\eta(u) := \eta(u - w_t)^T g_t + \eta^2 ((u - w_t)^T g_t)^2$$

Since surrogate is exp-concave for each fixed $\eta$, we can use online quasi-Newton method like Online Newton Step [Hazan et al., 2007] to get predictions $w_t^\eta$ that achieve logarithmic regret:

$$\sum_{t=1}^T \ell_t^\eta(w_t^\eta) - \sum_{t=1}^T \ell_t^\eta(u) \leq O(d \ln T) \quad \forall u \in \mathcal{U}$$

To learn the best $\eta$ we combine the predictions $w_t^\eta$ for multiple $\eta$ into a single master prediction $w_t$ using an experts algorithm for combining multiple learning rates similar to Squint [Koolen and Van Erven, 2015], to get:

$$\sum_{t=1}^T \ell_t^\eta(w_t^\eta) - \sum_{t=1}^T \ell_t^\eta(w_t^\eta) \leq O(\ln \ln T) \quad \forall \eta$$

**Difficulty:** Master has to perform well under multiple loss functions simultaneously. No standard experts algorithm works!

Together: $- \sum_{t=1}^T \ell_t^\eta(u) \leq O(d \ln T)$ for each $\eta$ and $u$, resulting in

$$R_T \leq \sum_{t=1}^T (w_t - u)^T g_t \leq \frac{O(d \ln T)}{\eta} + \eta V_t^u \Rightarrow O\left(\sqrt{V_t^u d \ln T}\right).$$
MetaGrad Algorithm
MetaGrad Algorithm

\[ \eta_1 \]  
\[ \Sigma_1 \]  
\[ w_1 \]

\[ \eta_2 \]  
\[ \Sigma_2 \]  
\[ w_2 \]

\[ \eta_3 \]  
\[ \Sigma_3 \]  
\[ w_3 \]

\[ \eta_4 \]  
\[ \Sigma_4 \]  
\[ w_4 \]

\[ \ldots \]  
\[ \ln(T) \]  
\[ \leq 16 \]
MetaGrad Algorithm

\[ \eta_1 \quad \eta_2 \quad \eta_3 \quad \eta_4 \]

\[ \Sigma_1 \quad \Sigma_2 \quad \Sigma_3 \quad \Sigma_4 \]

\[ w_1 \quad w_2 \quad w_3 \quad w_4 \]

\[ \ln(T) \leq 16 \]
MetaGrad Algorithm

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]

\[ \eta_1 \]
\[ \eta_2 \]
\[ \eta_3 \]
\[ \eta_4 \]

\[ \Sigma_1 \]
\[ w_1 \]

\[ \Sigma_2 \]
\[ w_2 \]

\[ \Sigma_3 \]
\[ w_3 \]

\[ \Sigma_4 \]
\[ w_4 \]

\[ \ln(T) \leq 16 \]
MetaGrad Algorithm

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]

\[ \sum \ln(T) \leq 16 \]
MetaGrad Algorithm

\[ \eta_1 \]
\[ \Sigma_1 \]
\[ w_1 \]

\[ \eta_2 \]
\[ \Sigma_2 \]
\[ w_2 \]

\[ \eta_3 \]
\[ \Sigma_3 \]
\[ w_3 \]

\[ \eta_4 \]
\[ \Sigma_4 \]
\[ w_4 \]

\[ \cdots \ln(T) \leq 16 \]

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]

\[ g = \nabla f(w) \]

\[ w \rightarrow \pi \]
MetaGrad Algorithm

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]

\[ \pi_i \leftarrow \pi_i e^{-\eta_i r_i - \eta_i^2 r_i^2} \]

where \( r_i = (w_i - w)^\top g \)

Tilted Exponential Weights
MetaGrad Algorithm

\[ \eta_1 \]
\[ \Sigma_1 \]
\[ \pi \]

\[ \eta_2 \]
\[ \Sigma_2 \]
\[ \pi \]

\[ \eta_3 \]
\[ \Sigma_3 \]
\[ \pi \]

\[ \eta_4 \]
\[ \Sigma_4 \]
\[ \pi \]

\[ \ldots \ln(T) \leq 16 \]

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]
\[ \pi_i \leftarrow \pi_i e^{-\eta_i r_i - \eta_i^2 r_i^2} \]
where \( r_i = (w_i - w)^\top g \)

Tilted Exponential Weights

\[ g = \nabla f(w) \]
MetaGrad Algorithm

\[ \eta_1 \]

\[ \eta_2 \]

\[ \eta_3 \]

\[ \eta_4 \]

\[ \cdots \]

\[ \ln(T) \leq 16 \]

\[ \Sigma_i \leftarrow (\Sigma_i^{-1} + 2\eta_i^2 gg^\top)^{-1} \]

\[ w_i \leftarrow w_i - \eta_i \Sigma_i g (1 + 2\eta_i r_i) \]

\[ \approx \text{Online Newton Step} \]

\[ \pi \]

\[ w = \frac{\sum_i \pi_i \eta_i w_i}{\sum_i \pi_i \eta_i} \]

\[ \pi_i \leftarrow \pi_i e^{-\eta_i r_i - \eta_i^2 r_i^2} \]

where \( r_i = (w_i - w)^\top g \)

Tilted Exponential Weights

\[ g = \nabla f(w) \]
MetaGrad Adapts to Curvature

MetaGrad regret bound:

\[ R_T = O\left(\sqrt{V_T d \ln T}\right) \]

**Corollary**

For \( \alpha \)-exp-concave or \( \alpha \)-strongly convex losses, MetaGrad ensures

\[ R_T = O\left(d \ln T\right) \]

without knowing \( \alpha \).
MetaGrad Adapts to Curvature

MetaGrad regret bound:

\[ R_T = O \left( \sqrt{V_T d \ln T} \right) \]

Corollary

For \( \alpha \)-exp-concave or \( \alpha \)-strongly convex losses, MetaGrad ensures

\[ R_T = O \left( d \ln T \right) \]

without knowing \( \alpha \).

Same result for fixed \( f_t = f \) (classical optimisation) even without curvature via derivative condition.
MetaGrad Adapts to Curvature

MetaGrad regret bound:

\[ R_T = O\left(\sqrt{V_T} d \ln T\right) \]

Corollary

For \( \alpha \)-exp-concave or \( \alpha \)-strongly convex losses, MetaGrad ensures

\[ R_T = O\left(d \ln T\right) \]

without knowing \( \alpha \).

Same result for fixed \( f_t = f \) (classical optimisation) even without curvature via derivative condition.

Reason

Curvature implies \( \Omega(V_T) \) cumulative slack between loss and its tangent lower bound.
MetaGrad Adapts to Stochastic Margin

Consider i.i.d. losses $f_t \sim \mathbb{P}$ with **stochastic optimum**

$$u^* = \arg \min_u \mathbb{E} f(u)$$

Goal is small **pseudo-regret** compared to $u^*$:

$$R_T^* = \sum_{t=1}^T f_t(w_t) - \sum_{t=1}^T f_t(u^*)$$

Corollary (with Peter Grünwald)

For any $\beta$-Bernstein $\mathbb{P}$, MetaGrad keeps the expected regret below

$$\mathbb{E} R_T^* \leq O \left( \frac{d \ln T}{2} - \beta T \frac{1}{2} - \beta^2 \right).$$

Fast rates without curvature: e.g. absolute loss, hinge loss, ... Reason Bernstein bounds $\mathbb{E} V_T^*$ above by $\mathbb{E} R_T^*$. “Solve” regret bound.
MetaGrad Adapts to Stochastic Margin

Consider i.i.d. losses $f_t \sim \mathbb{P}$ with stochastic optimum

$$u^* = \arg \min_u \mathbb{E} f(u)$$

Goal is small pseudo-regret compared to $u^*$:

$$R_T^* = \sum_{t=1}^T f_t(w_t) - \sum_{t=1}^T f_t(u^*)$$

Corollary (with Peter Grünwald)

For any $\beta$-Bernstein $\mathbb{P}$, MetaGrad keeps the expected regret below

$$\mathbb{E} R_T^* \leq O \left( (d \ln T)^{\frac{1}{2-\beta}} T^{\frac{1-\beta}{2-\beta}} \right).$$

Fast rates without curvature: e.g. absolute loss, hinge loss, ...
MetaGrad Adapts to Stochastic Margin

Consider i.i.d. losses \( f_t \sim \mathbb{P} \) with stochastic optimum

\[
u^* = \arg \min_u \mathbb{E} f(u)\]

Goal is small pseudo-regret compared to \( u^* \):

\[
R^*_T = \sum_{t=1}^{T} f_t(w_t) - \sum_{t=1}^{T} f_t(u^*)
\]

Corollary (with Peter Grünwald)

For any \( \beta \)-Bernstein \( \mathbb{P} \), MetaGrad keeps the expected regret below

\[
\mathbb{E} R^*_T \leq O \left( (d \ln T)^{\frac{1}{2-\beta}} T^{\frac{1-\beta}{2-\beta}} \right).
\]

Fast rates without curvature: e.g. absolute loss, hinge loss, ...

Reason

Bernstein bounds \( \mathbb{E}[V^*_T] \) above by \( \mathbb{E}[R^*_T] \). “Solve” regret bound.
Experiments

(a) Offline: \( f_t(u) = |u - 1/4| \)

(b) Stochastic Online: \( f_t(u) = |u - x_t| \) where \( x_t = \pm \frac{1}{2} \) i.i.d. with probabilities 0.4 and 0.6.

Figure: Examples of fast rates on functions without curvature. MetaGrad incurs logarithmic regret \( O(\log T) \), while AdaGrad incurs \( O(\sqrt{T}) \) regret, matching its bound.
Conclusion

First contact with a new generation of adaptive algorithms.
Conclusion

First contact with a new generation of adaptive algorithms.

MetaGrad adapts to a wide range of environments:

- Stochastic data: $\frac{1-\beta}{2-\beta}$
- Curvature: $d \ln T$
- Worst case: $\sqrt{T}$