UNIVERSITY of WASHINGTON

Differentiable Programming à la Moreau

Vincent Roulet, Zaid Harchaoui
University of Washington

Idea

- Smoothing an objective by a Moreau envelope can enhance the capabilities of gradient-based methods (Nesterov (2005), Lin et al (2018))
- But computing the Moreau envelope may be as expensive as minimizing the objective...
- How to exploit the computational structure of an objective to approximate a Moreau envelope in a differentiable programming framework?

Moreau Gradient

Overview Consider an objective f

- Gradient-based algo. use linear approx. of f
  $\rightarrow$ oracle accuracy *fixed* by smoothness prop. of f
- Algo. based on Moreau-envelope use regularized min. of f
  $\rightarrow$ oracle accuracy *controlled* by optimization subroutine

Moreau Envelope of αf on x

$$\mathrm{env}(\alpha f)(x) = \inf_{y \in \mathbb{R}^d} \alpha f(x-y) + \|y\|_2^2/2$$

well-defined for $0 \leq \alpha < \bar{\alpha}$ s.t. $y \mapsto \bar{\alpha} f(x-y) + \|y\|_2^2/2$ is convex

Moreau Gradient of f on x with stepsize $0 \leq \alpha < \bar{\alpha}$

$$\nabla \mathrm{env}(\alpha f)(x) = \arg\min_{y \in \mathbb{R}^d} \alpha f(x-y) + \|y\|_2^2/2$$

- Maximal stepsize $\bar{\alpha}$ larger than gradient descent stepsize
- Necessary optimal cond.: $x^* \in \arg\min_x f(x) \Rightarrow \nabla \mathrm{env}(\alpha f)(x^*) = 0$
- Generally not available in closed form

Approximate Moreau Gradient Optimization

$$x^{(k+1)} = x^{(k)} - \widehat{\nabla} \mathrm{env}(\alpha f)(x^{(k)})$$

for $\widehat{\nabla} \mathrm{env}(\alpha f)(x) \approx \nabla \mathrm{env}(\alpha f)(x)$

- Direct implementation:

$$\widehat{\nabla} \mathrm{env}(\alpha f)(x) = \mathcal{A}_k \left(\alpha f(x - \cdot) + \|\cdot\|_2^2/2\right)$$

  for $\mathcal{A}_k(h)$ the k^{th} iterate of algo. $\mathcal{A}$ on h such as gradient descent
- Here: Implement f in a differentiable programming framework that gives access to Moreau gradients in a backward pass like

```
y = func(x)
m_grad = auto_m_grad(y, x, alpha)
```

  with m_grad$=\nabla \mathrm{env}(\alpha f)(x)$ computed from graph of comput. of f.

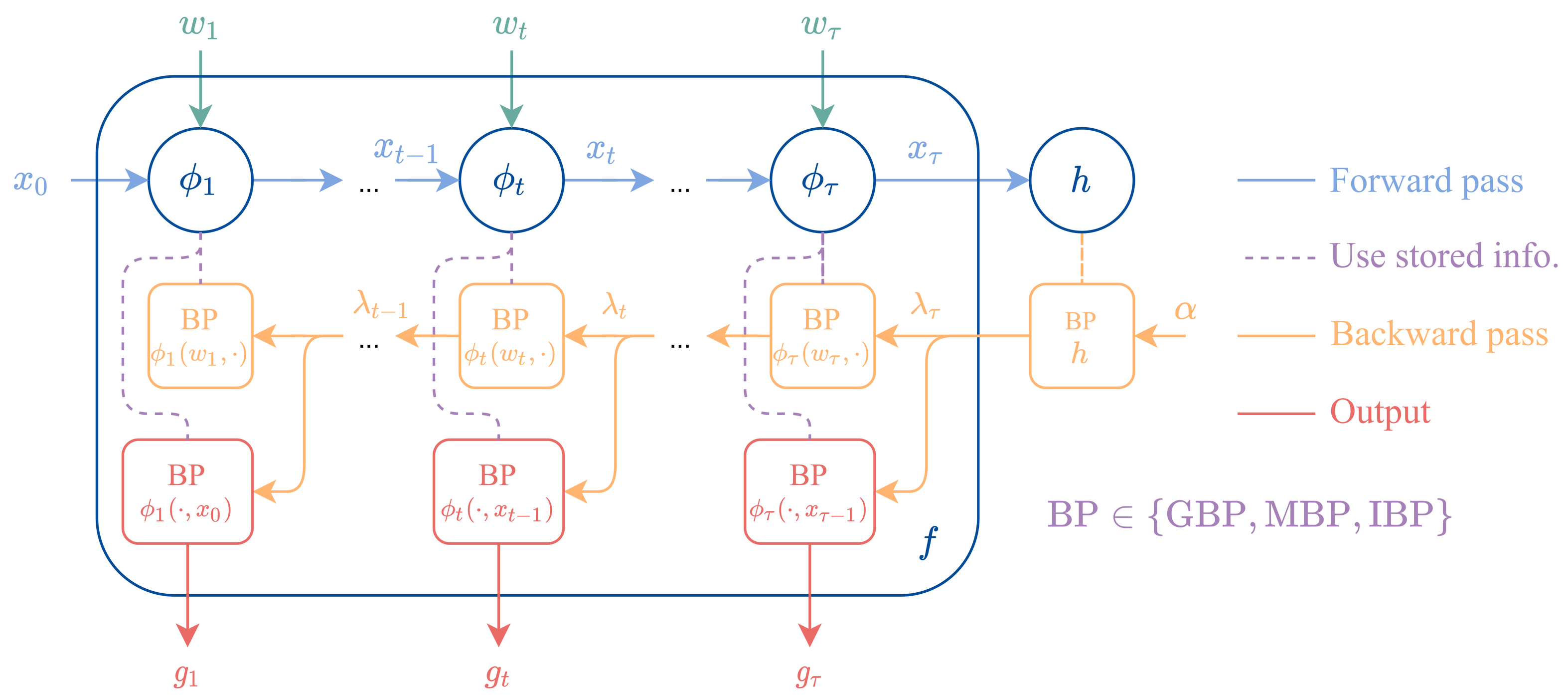
Differentiable Programming

Consider a function f with a dynamical structure

$$f(\boldsymbol{w}) = x_\tau,$$
$$\text{s.t. } x_t = \phi_t(w_t, x_{t-1}) \text{ for } t = 1, \ldots, \tau, \boldsymbol{w} = (w_1, \ldots, w_\tau)$$

for x_0 fixed as in, e.g., deep learning or nonlinear control.

Differentiable Programming of an objective $h \circ f$



1. In a forward pass, compute f step by step through the functions ϕ_t, store the intermediate computations ϕ_t with their inputs x_t, w_t
2. In a backward pass, back-propagate co-state variables λ_t using one of the following back-propagation rule BP on $\phi_t(w_t, \cdot)$ or $\phi_t(\cdot, x_t)$ starting from $\lambda_\tau = \mathrm{BP}(h)(x_\tau, \alpha)$
   - $\mathrm{GBP}(f)(x, \lambda) = \nabla f(x) \lambda$
     $\rightarrow$ classical back-propagation rule used in auto.-diff.
   - $\mathrm{MBP}(f)(x, \lambda) = \arg\min_y \lambda^\top f(x-y) + \|y\|_2^2/2$
     $\rightarrow$ generalized Moreau gradient for multivariate function
   - $\mathrm{IBP}(f)(x, \lambda) = \arg\min_y \|f(x-y) - f(x) + \lambda\|_2^2 + \|y\|_2^2/2$
     $\rightarrow$ regularized inverse as in target propagation Lee et al (2015)
3. Plug output oracle directions $(g_t)_{t=1}^\tau$ in optimizer like SGD, Adam, ...

Note: Can mix BP procedures such as using

$\mathrm{GBP}(h)$, $\mathrm{GBP}(\phi_t(w_t, \cdot))$, $\mathrm{IBP}(\phi_t(\cdot, x_{t-1}))$ as Frerix et al (2018)

Implementation

Use k iterations of algo. $\mathcal{A}$ such as grad.descent to approx. BP such as

$$\mathrm{MBP}(f)(x, \lambda) \approx \mathcal{A}_k(\lambda^\top f(x - \cdot) + \|\cdot\|_2^2/2)$$

Overall complexity of oracle: k times more than classical backprop.

1 Forward pass

Inputs: Function f parameterized by $(\phi_t)_{t=1}^\tau$, input x_0, param. $(w_t)_{t=1}^\tau$
for $t = 1, \ldots, \tau$ **do**
  Compute $x_t = \phi_t(w_t, x_{t-1})$
  Store x_{t-1}, w_t, ϕ_t
end for
Output: Function eval. x_τ
Stored: Comput. $(x_{t-1}, w_t, \phi_t)_{t=1}^\tau$

2 Backward pass

Inputs: Stored $(x_{t-1}, w_t, \phi_t)_{t=1}^\tau$, output x_τ, objective h, stepsize α
Initialize $\lambda_\tau = \mathrm{BP}(h)(x_\tau, \alpha)$
for $t = \tau, \ldots, 1$ **do**
  Get $\lambda_{t-1} = \mathrm{BP}(\phi_t(w_t, \cdot))(x_t, \lambda_t)$
  Get $g_t = \mathrm{BP}(\phi_t(\cdot, x_{t-1}))(w_t, \lambda_t)$
end for
Output: Oracle directions $(g_t)_{t=1}^\tau$.

Chain Rule

Moreau Gradient Rule for composition $h \circ f$

Under suitable assumptions, comput. of Moreau gradient decomposes as

$$\nabla \mathrm{env}(\alpha h \circ f)(x) = \arg\min_y \left\{ \lambda^{*\top} f(x-y) + \|y\|_2^2/2 \right\}$$
$$\text{where} \quad \lambda^* = \arg\max_\lambda -(\alpha h)^*(\lambda) + \mathrm{env}(\lambda^\top f)(x)$$

- Proximal grad. step to compute λ^* gives MBP rule:
  $\rightarrow \nabla \mathrm{env}(\alpha h \circ f)(x) \approx \nabla \mathrm{env}(\lambda^\top f)(x)$ for $\lambda = \nabla \mathrm{env}(\alpha h)(f(x))$

Regularized Inverse Rule for composition $h \circ f$

Comput. of Moreau gradient amounts to solve

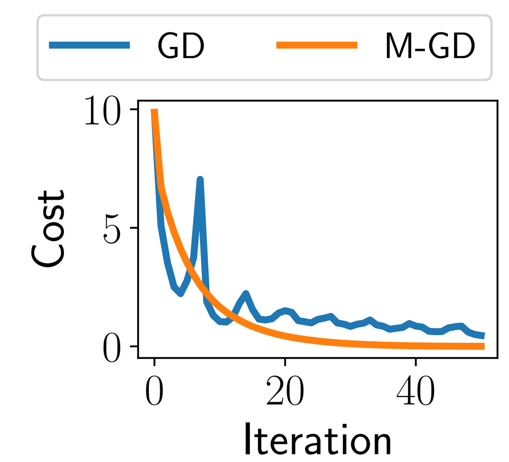
$$\min_\lambda \alpha f(g(x) - \lambda) + p(\lambda) \text{ for } p(\lambda) = \min\left\{\|y\|_2^2/2 : g(x) - g(x-y) = \lambda\right\}$$

- Incremental proximal point to compute λ^* gives IBP rule:
  $\rightarrow \nabla \mathrm{env}(\alpha h \circ f)(x) \approx \mathrm{IBP}(f)(x; \lambda)$ for $\lambda = \nabla \mathrm{env}(\alpha h)(f(x))$

Experiments

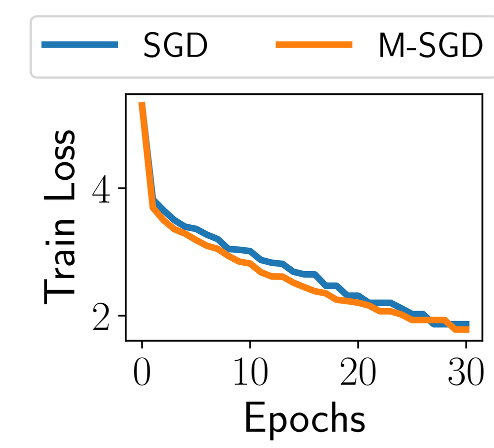
Moreau Gradient Descent (M-GD)

- Nonlinear control: swinging up pendulum
- Use approx. Moreau grad. on output of deterministic dynamical system



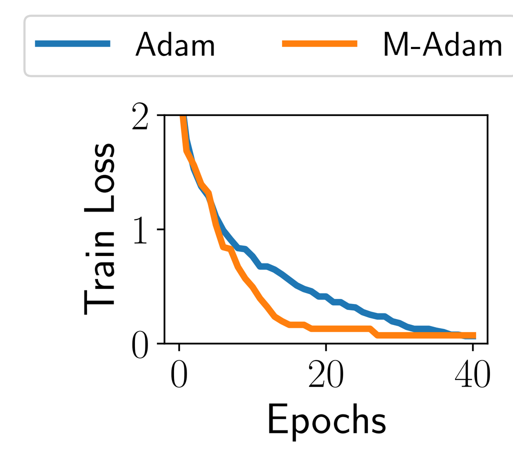
Stoch. Moreau Grad. Desc. (M-SGD)

- MLP on CIFAR10
- Compute oracles on mini-batches S, i.e., $\widehat{\nabla} \mathrm{env}(\alpha F_S)$ for $F_S(w) = \sum_{i \in S} f_i(w)$



Adam with Moreau Grad. (M-Adam)

- AllCNN ConvNet on CIFAR10
- Compute oracles on mini-batches S, i.e., $\widehat{\nabla} \mathrm{env}(\alpha F_S)$ for $F_S(w) = \sum_{i \in S} f_i(w)$
- Plug oracle directions in Adam optimizer



References

Nesterov, Y. (2005). Smooth Minimization of Non-Smooth Functions. *Mathematical Programming.*
Hongzhou, L., Mairal, J., Harchaoui Z. (2018). Catalyst Acceleration for First-order Convex Optimization: from Theory to Practice. *Journal of Machine Learning Research.*
Lee, D., Zhang, S., Fischer, A., Bengio, Y. (2015). Difference Target Propagation. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.*
Frerix, T., Möllenhoff, T., Moeller, M., Cremers, D. (2018). Proximal Backpropagation. *International Conference on Learning Representations.*