CSC 2541: Neural Net Training Dynamics

Lecture 9 - Momentum

Roger Grosse

University of Toronto, Winter 2022

Today

- So far, we've focused entirely on gradient descent dynamics
- In the remainder of the course, we'll branch out to more general dynamics
- Today
  - What can happen in more general dynamical systems?
  - Momentum optimization
    - understanding your homework derivation
    - Nesterov Accelerated Gradient
    - accelerated convergence
  - A brief taste of what we can learn from linear systems and control theory.
- Lots of similar ideas used to understand differentiable game dynamics

Dynamical Systems

- So far, all of our analysis has been based on a single recurrence:

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \alpha \nabla \mathcal{J}(\mathbf{w}^{(k-1)})$$

(preconditioning = GD in another coordinate system)

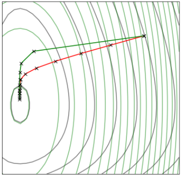
- Linear case ($\mathbf{H}$ symmetric):

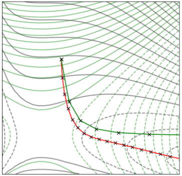
$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \alpha \mathbf{H} \mathbf{w}^{(k-1)} \qquad \Rightarrow \qquad \mathbf{w}^{(k)} = (\mathbf{I} - \alpha \mathbf{H})^k \mathbf{w}^{(0)}$$

- Gradient flow

$$\dot{\mathbf{w}} = -\alpha \nabla \mathcal{J}(\mathbf{w}) = -\alpha \mathbf{H} \mathbf{w} \qquad \Rightarrow \qquad \mathbf{w}(t) = \exp(-\alpha t \mathbf{H}) \mathbf{w}(0)$$

- What can happen?





Dynamical Systems

- Now let's move beyond this and consider other sorts of dynamics
  - **Today:** momentum (accelerating convergence for ordinary optimization)
  - **Next week:** simultaneous optimization for differential games
  - **Weeks 11 and 12:** bilevel optimization
- Consider more general dynamics
  - **Discrete time:**
$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \alpha \mathbf{f}(\mathbf{w}^{(k-1)})$$
  - **Continuous time:**
$$\dot{\mathbf{w}} = -\alpha \mathbf{f}(\mathbf{w})$$
- $\mathbf{f}$ is a vector field which is not necessarily integrable, i.e. not necessarily the gradient of any function

Dynamical Systems

- Linear case:
  - $\mathbf{f}(\mathbf{w}) = \mathbf{A}\mathbf{w}$, with $\mathbf{A}$ not necessarily symmetric
  - **Discrete time:**

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \alpha \mathbf{A}\mathbf{w}^{(k-1)} \qquad \Rightarrow \qquad \mathbf{w}^{(k)} = (\mathbf{I} - \alpha \mathbf{A})^k \mathbf{w}^{(0)}$$

  - **Continuous time:**

$$\dot{\mathbf{w}} = -\alpha \mathbf{A}\mathbf{w} \qquad \Rightarrow \qquad \mathbf{w}(t) = \exp(-\alpha t \mathbf{A})\mathbf{w}(0)$$

- If $\mathbf{A}$ is not symmetric, it can have complex and/or repeated eigenvalues. This leads to more possible behaviors:

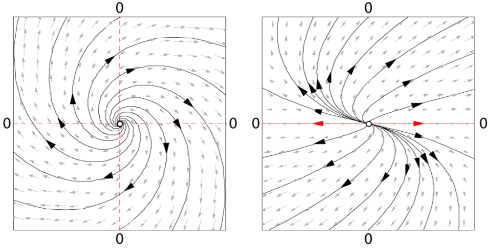


Image: https://en.wikipedia.org/wiki/Dynamical_system

Dynamical Systems

What else can happen in the nonlinear case?

- In 2 dimensions or higher, you can get limit cycles:

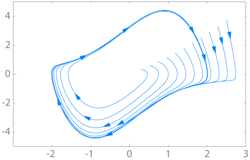
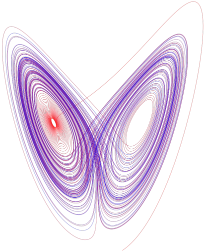


Image: By User:XaosBits at English Wikipedia, CC BY 2.5, https://commons.wikimedia.org/w/index.php?curid=732841

- In 3 dimensions or higher, you can get chaotic dynamics, such as strange attractors. Here's the Lorenz system:

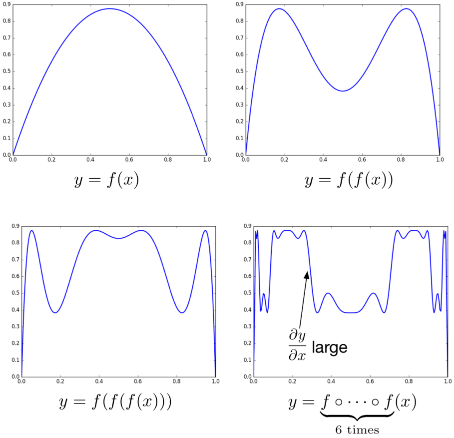


$$\dot{x} = a(y - x)$$
$$\dot{y} = x(b - z) - y$$
$$\dot{z} = xy - cz$$

Image: By Dschwen - Own work, CC BY 2.5, https://commons.wikimedia.org/w/index.php?curid=494694

For discrete mappings, chaos can arise even more easily.

$$f(x) = 3.5\,x\,(1 - x)$$



$$f_c(z) = z^2 + c$$

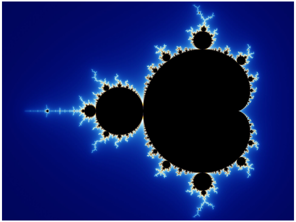


Image: `https://en.wikipedia.org/wiki/Mandelbrot_set`

Heavy Ball Momentum

Heavy Ball Momentum

- Heavy ball momentum is a simple and highly effective method for speeding up convergence of gradient descent

$$\mathbf{v}^{(k+1)} \leftarrow \beta \mathbf{v}^{(k)} - \alpha \nabla \mathcal{J}(\mathbf{w}^{(k)})$$
$$\mathbf{w}^{(k+1)} \leftarrow \mathbf{w}^{(k)} + \mathbf{v}^{(k+1)}$$

- α is the learning rate, just like in gradient descent.
- β is a damping/viscosity parameter. It should be slightly less than 1 (e.g. 0.9 or 0.99). Why not exactly 1?
- Continuous dynamics (ignore learning rate for simplicity):

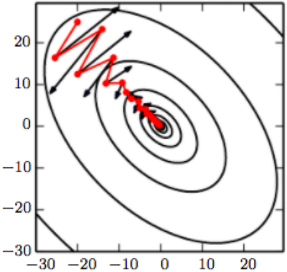
$$\dot{\mathbf{v}}(t) = -\mu \mathbf{v}(t) - \nabla \mathcal{J}(\mathbf{w}(t))$$
$$\dot{\mathbf{w}}(t) = \mathbf{v}(t)$$

- **Physical analogy:** imagine a “heavy ball” rolling on a nearly flat surface, where $\mathcal{J}$ represents height

Why is this a good idea?

- No one-sentence explanation that I'm aware of
- Ordinary gradient descent corresponds to $\beta = 0$ (extremely high damping/viscosity). This is like submerging the ball in a thick fluid

- In the high curvature directions, the gradients cancel each other out, so momentum dampens the oscillations.
- In the low curvature directions, the gradients point in the same direction, allowing the parameters to pick up speed.



Goodfellow et al., *Deep Learning*

- For homework, you analyzed its convergence in the quadratic case by computing the system's eigenvalues. Let's try to understand why you got the answer that you did.

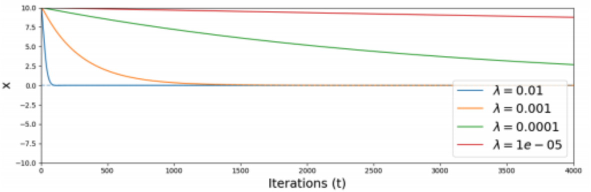
Heavy Ball Momentum

In the problem set, you analyzed the dynamics of HB for convex quadratics. Recap:

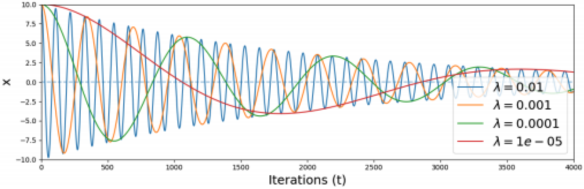
- Fixed points: $\nabla \mathcal{J}(\mathbf{w}) = \mathbf{0}, \mathbf{v} = \mathbf{0}$
- Rotation invariant
- Can assume diagoal WLOG, in which case each coordinate evolves independently
- There's a critical threshold T such that directions with $0 < h_j < T$ approach 0 monotonically (the overdamped case) and directions with $T < h_j < h_{\max}$ oscillate (the underdamped case)
  - Underdamped directions have only real eigenvalues, while overdamped directions have complex eigenvalues
  - $T = \alpha^{-1}(1 - \sqrt{\beta})^2 = \mathcal{O}(\alpha^{-1}(1 - \beta)^2)$

Heavy Ball Momentum

- Dynamics of different eigendirections with $\beta = 0.9$ (all directions overdamped)



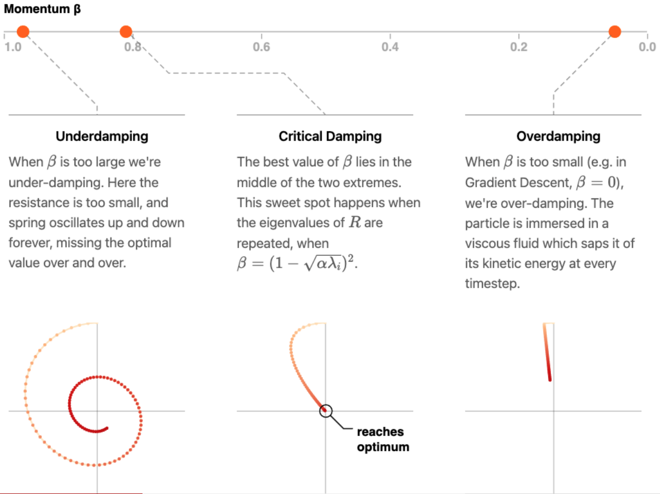
- And here's $\beta = 0.999$ (all directions underdamped)



- Figures from Lucas et al., "Aggregated momentum: Stability through passive damping"

Heavy Ball Momentum

- Phase space visualization (plots both w and v) from Goh, "Why momentum really works" (https://distill.pub/2017/momentum/)



Heavy Ball Momentum

The overdamped case:

- If the gradient is constant (i.e. the cost surface is a plane), the parameters will reach a terminal velocity of

$$-\frac{\alpha}{1-\beta}\nabla\mathcal{J}(\mathbf{w}),$$

  which resembles gradient descent with learning rate $\tilde{\alpha} = \alpha/(1-\beta)$. This quantity is the effective learning rate.
- If $\tilde{\alpha}h$ is very small (the highly overdamped case), the particle will move slowly, and this should be a good approximation.
- For a convex quadratic, the spectral radius for SGD with learning rate $\tilde{\alpha}$ is $|1-\tilde{\alpha}h|$.
- In your homework, you probably derived an answer like:

$$\tfrac{1}{2}(\gamma + \sqrt{\gamma^2 - 4\beta}) \qquad \gamma = 1 + \beta - \alpha h$$

  With a bunch of algebra, you can show this is approximately $1 - \tilde{\alpha}h$ for small α. (Try looking at the limit as $\alpha \to 0$.)

Heavy Ball Momentum

The underdamped case:

- Let's start with the continuous dynamics:

$$\begin{aligned}\dot{\mathbf{v}}(t) &= -\mu \mathbf{v}(t) - \nabla \mathcal{J}(\mathbf{w}(t)) \\ \dot{\mathbf{w}}(t) &= \mathbf{v}(t)\end{aligned}$$

- A common way to prove stability of a dynamical system is to find a Lyapunov function, which is nonincreasing and is minimized at the equlibrium point
- For systems based on physics, this is often related to the energy
- Define

$$\mathcal{E} = \underbrace{\mathcal{J}(\mathbf{w})}_{\text{potential energy}} + \underbrace{\tfrac{1}{2}\|\mathbf{v}\|^2}_{\text{kinetic energy}}$$

- Change in energy over time (i.e. dissipation):

$$\begin{aligned}\dot{\mathcal{E}} &= \dot{\mathbf{w}}^\top \nabla \mathcal{J}(\mathbf{w}) + \dot{\mathbf{v}}^\top \mathbf{v} \\ &= \mathbf{v}^\top \nabla \mathcal{J}(\mathbf{w}) - \nabla \mathcal{J}(\mathbf{w})^\top \mathbf{v} - \mu \mathbf{v}^\top \mathbf{v} \\ &= -\mu \|\mathbf{v}\|^2\end{aligned}$$

Heavy Ball Momentum

- Consider the dynamics for a convex quadratic along one eigenvector with curvature h
- Suppose there's no damping, i.e. $\mu = 0$. Then energy is conserved.
- Eliminating v, we can rewrite the dynamics as

$$\ddot{w}(t) = -hw$$

- This is a simple harmonic oscillator. If $w(0) > 0$ and $v(0) = 0$, then it has the solution

$$\begin{aligned} w(t) &= A \cos \omega t \\ v(t) = \dot{w}(t) &= -\omega A \sin \omega t \\ A &= w(0) \\ \omega &= \sqrt{h} \end{aligned}$$

- Observe that $\mathcal{E} = \frac{1}{2}\omega^2 A^2 = \frac{1}{2} h A^2$

Heavy Ball Momentum

- Now suppose μ is small. There are two timescales:
  - On the short timescale, it's a harmonic oscillator with amplitude $A(t)$
  - On the long timescale, energy dissipates
- Instantaneous dissipation:

$$\dot{\mathcal{E}}(t) = -\mu \|\mathbf{v}(t)\|^2 = -\mu\omega^2 A(t)^2 \sin^2 \omega t$$

- On a long timescale, the rate of dissipation is the temporal average of $\dot{\mathcal{E}}$, which is

$$-\tfrac{1}{2}\mu\omega^2 A(t)^2 = -\tfrac{1}{2}\mu h A(t)^2 = -\mu\mathcal{E}(t)$$

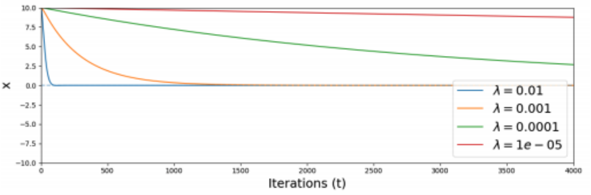
- Differential equation:

$$\dot{\mathcal{E}}(t) = -\mu\mathcal{E},$$

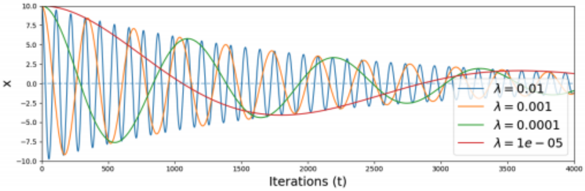
  which is exponential decay with timescale $1/\mu$
- Note: this is independent of h!

Heavy Ball Momentum

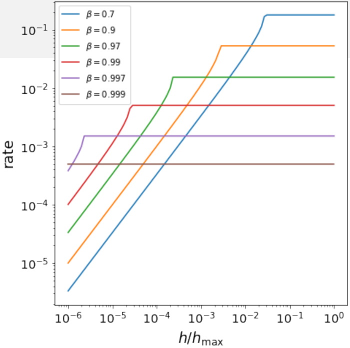
- Compare to the observed behavior
- $\beta = 0.9$ (overdamped):



- $\beta = 0.999$ (underdamped):



Heavy Ball Momentum



- For homework, you derived the spectral radius

$$\rho = \begin{cases} \frac{1}{2}(\gamma + \sqrt{\gamma^2 - 4\beta}) & \text{if } h \le T \\ \sqrt{\beta} & \text{if } h > T. \end{cases}$$

$$T = \alpha^{-1}(1 - \sqrt{\beta})^2$$

- This is plotted on the right (assuming $\alpha = h_{\max}^{-1}$)
- Based on this figure, you want to choose β such that the minimum curvature direction is critically damped, i.e., $T = h_{\min}$
- Solving for β,

$$\beta = \left(1 - \frac{1}{\sqrt{\kappa}}\right)^2$$

- Rate of convergence (all directions are underdamped):

$$-\log \rho = -\tfrac{1}{2} \log \beta \approx 1/\sqrt{\kappa}$$

- Compare to $1/\kappa$ for gradient descent

Heavy Ball Momentum

An aside:

- We analyzed the underdamped case for the continuous dynamics (damped harmonic oscillator).
  - Why does this predict the behavior in the discrete case? Shouldn't the discretization error hurt convergence?
- In particular, if $\beta = 1$, then the spectral radius is 1.
- In the continuous case, we explained this using conservation of energy. Does this extend to the discrete dynamics?
  - **No!** Energy is not conserved!
  - The actual reason is very deep.
- I noticed this puzzle when typing up the homework solutions. Thanks to Chris Maddison for pointing me to the answer!

Heavy Ball Momentum

- By shifting the "start" of the update by half a time step, we can rewrite HB momentum as a leapfrog integrator, a kind of symplectic integrator:

$$
\begin{aligned}
\mathbf{v}^{(k+\frac{1}{2})} &= \mathbf{v}^{(k)} - \frac{\alpha}{2}\nabla\mathcal{J}(\mathbf{w}^{(k)}) \\
\mathbf{w}^{(k+1)} &= \mathbf{w}^{(k)} + \mathbf{v}^{(k+\frac{1}{2})} \\
\mathbf{v}^{(k+1)} &= \mathbf{v}^{(k+\frac{1}{2})} - \frac{\alpha}{2}\nabla\mathcal{J}(\mathbf{w}^{(k+1)})
\end{aligned}
$$

- Symplectic integrators can be shown to approximately conserve a different but related function called the shadow Hamiltonian. For quadratics, it's exact. See Hairer et al., "Geometric numerical integration"

Nesterov Accelerated Gradient

Nesterov Accelerated Gradient

- Polyak invented HB momentum in 1964 (and discussed the physics analogy)
- Nesterov invented a similar update rule in 1983 now called the Nesterov Accelerated Gradient (NAG) which he proved achieved optimal convergence for convex quadratics
  - Even though Nesterov was Polyak's student, he seems not to have mentioned the physics analogy
- Methods similar to HB and NAG are often called accelerated methods. Ironically, the term "accelerated" has nothing to do with the physics analogy and just refers to converging faster.
  - "Chebyshev acceleration" predated the HB paper by about a decade
- Sutskever et al. (2013) popularized NAG in machine learning and revived the momentum interpretation

Nesterov Accelerated Gradient

- NAG update rule (as presented in d'Aspremont et al., "Accelerated Methods"):

$$\begin{aligned} \mathbf{y}^{(k)} &= \mathbf{w}^{(k)} + \tau_k(\mathbf{z}^{(k)} - \mathbf{w}^{(k)}) \\ \mathbf{w}^{(k+1)} &= \mathbf{y}^{(k)} - \alpha_k \nabla \mathcal{J}(\mathbf{y}^{(k)}) \\ \mathbf{z}^{(k+1)} &= \mathbf{z}^{(k)} - \gamma_k \nabla \mathcal{J}(\mathbf{y}^{(k)}) \end{aligned}$$

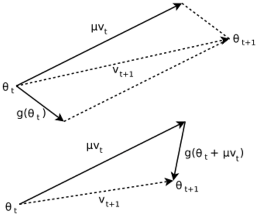
- Nesterov used a carefully chosen schedule of α_k, τ_k, and γ_k to obtain optimal convergence rates. In DL, we tend to use constant values.

Nesterov Accelerated Gradient

- Sutskever et al. (2013) rewrote the update in a way that emphasizes its similarity to HB:

$$\mathbf{v}^{(k+1)} = \beta \mathbf{v}^{(k)} - \alpha \nabla \mathcal{J}(\mathbf{w}^{(k)} + \beta \mathbf{v}^{(k)})$$
$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \mathbf{v}^{(k+1)}$$

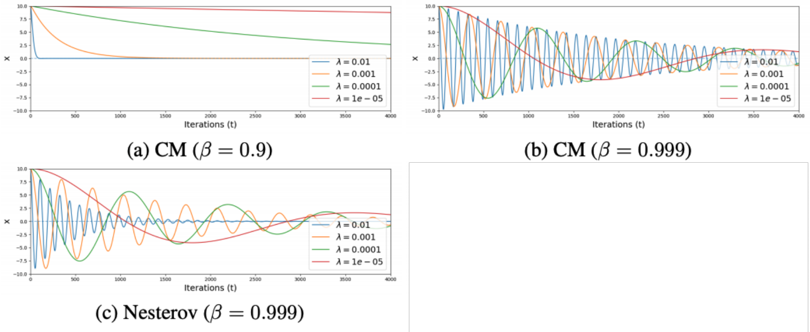
- This extrapolation trick is commonly used to dampen oscillations in dynamical systems. Some other examples:
  - The D term in PID controllers can be interpreted as extrapolating the state, and can dampen the oscillations created by the I term.
    - See Hu and Lessard, 2017, "Control interpretations for first-order optimization methods"
  - Extragradient, an algorithm for solving differentiable games (Lecture 10)



Top: HB momentum.
Bottom: NAG
(Sutskever et al., 2013)

Nesterov Accelerated Gradient

- Compared to HB, NAG dampens the high-frequency oscillations.



(a) CM ($\beta = 0.9$)

(b) CM ($\beta = 0.999$)

(c) Nesterov ($\beta = 0.999$)

- Figure from Lucas et al., "Aggregated momentum". CM = classical momentum (= HB).
- This effect doesn't affect the convergence rate for quadratics (since the low frequency oscillations come to dominate), but I'd guess this is helpful for non-quadratics (where high-frequency oscillations might cause bigger problems)

Aggregated Momentum

Lucas et al., ICML 2019, "Aggregated momentum: Stability through passive damping"

- Inspired by the idea of dampening oscillations, we came up with Aggregated Momentum (AggMo).
- This uses the heavy ball update, except that it additively combines N_V velocity vectors with different damping parameters.

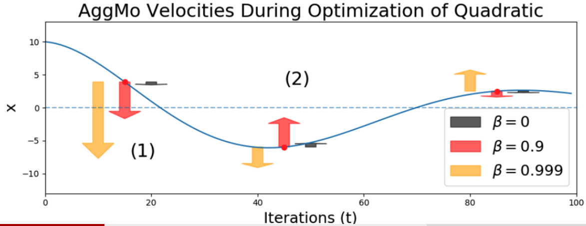
$$\mathbf{v}_i^{(k+1)} = \beta_i \mathbf{v}_i^{(k)} - \alpha \nabla \mathcal{J}(\mathbf{w}^{(k)}) \qquad \text{for } i = 1, \ldots, N_V$$

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \frac{1}{N_V} \sum_{i=1}^{N_V} \mathbf{v}_i^{(k+1)}$$

- Reasonable default: $\boldsymbol{\beta} = [0, 0.9, 0.99]$
- NAG with fixed damping parameter $\tilde{\beta}$ is very nearly equivalent to AggMo with two velocity components, and $\boldsymbol{\beta} = [0, \tilde{\beta}]$. (Details in the paper — it's just a few lines of algebra.) So AggMo may provide a useful perspective on what NAG is doing.

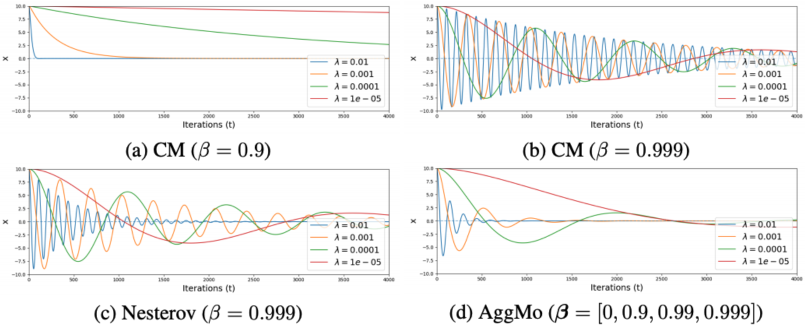
Aggregated Momentum

- Intuition: suppose $\boldsymbol{\beta} = [0, 0.9, 0.999]$.
- The velocity vector with damping parameter 0.999 will be the most aggressive, and generally the largest in magnitude.
- The velocity vector with parameter 0 points opposite the gradient direction, i.e. towards 0.
    - It will therefore tend to dissipate energy (by reducing the potential energy in each step.)
    - However, it is generally small in magnitude, so the dissipation effect is small.
- The velocity vector with parameter 0.9 will also tend to point inwards and hence dissipate energy. But it's larger in magnitude, and therefore has a stronger effect.



Aggregated Momentum

AggMo dampens oscillations even more strongly than NAG:



(a) CM ($\beta = 0.9$)

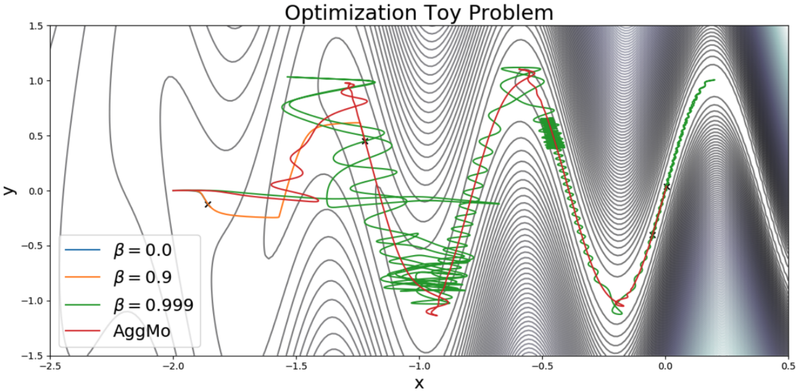
(b) CM ($\beta = 0.999$)

(c) Nesterov ($\beta = 0.999$)

(d) AggMo ($\boldsymbol{\beta} = [0, 0.9, 0.99, 0.999]$)

Aggregated Momentum

The dampening effect seems useful even outside of quadratic problems:



Accelerated Convergence

Accelerated Convergence

- Recall:
  - Condition number $\kappa = h_{\max}/h_{\min}$
  - Gradient descent on a convex quadratic requires $\mathcal{O}(\kappa)$ iterations to reach a given loss (Lecture 1)
  - Conjugate gradient requires $\mathcal{O}(\sqrt{\kappa})$ iterations
  - In the problem set, you showed much faster convergence was possible using HB
- How fast do HB and NAG converge?
- Can we do better, e.g. with a fancier update rule, or using more than 2 past iterates?

Accelerated Convergence

- Oracle model of optimization: in each iteration, you query a weight vector $\mathbf{w}^{(k)}$, and the oracle returns $\mathcal{J}(\mathbf{w}^{(k)})$ and $\nabla\mathcal{J}(\mathbf{w}^{(k)})$.
  - Think of the oracle as an adversary (it can choose values and gradients to make life hard for your algorithm)
  - You can't query things like the sparsity pattern, curvature, etc., so this rules out preconditioning
  - Captures iterative methods like GD, HB, NAG, CG
- Strongly convex optimization:
  - Strong convexity: for all $\mathbf{w}$ and $\mathbf{w}'$, and a parameter μ,

$$\mathcal{J}(\mathbf{w}) \geq \mathcal{J}(\mathbf{w}') + \nabla\mathcal{J}(\mathbf{w}')^\top(\mathbf{w} - \mathbf{w}') + \frac{\mu}{2}\|\mathbf{w} - \mathbf{w}'\|^2$$

  - Lipschitz smoothness: for all $\mathbf{w}$ and $\mathbf{w}'$ and a parameter L,

$$\|\nabla\mathcal{J}(\mathbf{w}) - \nabla\mathcal{J}(\mathbf{w}')\| \leq L\|\mathbf{w} - \mathbf{w}'\|$$

  - Condition number $\kappa = L/\mu$ (generalizes the quadratic case)

Accelerated Convergence

- The following function helps illustrate the difficulties of first-order optimization:

$$\mathcal{J}(\mathbf{w}) = \frac{1}{2}(1 - w_1)^2 + \sum_{j=1}^{D-1} \frac{1}{2}(w_{j+1} - w_j)^2$$

- **Observe:** This is a quadratic objective $\mathcal{J}(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{A}\mathbf{w} - \mathbf{b}^\top \mathbf{w}$, with (for $D = 5$):

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix} \qquad \mathbf{b} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

- **Initialization:** $w_1 = \cdots = w_D = 0$
- **Optimal solution:** $w_1 = \cdots = w_D = 1$
- Variants of this function can be used to show an $\mathcal{O}(\sqrt{\kappa})$ lower bound for convergence under the oracle model. See Nesterov, "Introductory lectures on convex optimization"

Accelerated Convergence

$$\mathcal{J}(\mathbf{w}) = \frac{1}{2}(1 - w_1)^2 + \sum_{j=1}^{D-1} \frac{1}{2}(w_{j+1} - w_j)^2$$

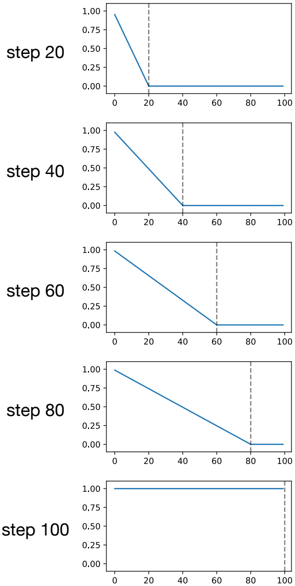
- Recall about conjugate gradient:
  - Krylov subspace

$$\mathcal{K}_k(\mathbf{A}, \mathbf{r}) = \text{span}\{\mathbf{r}, \mathbf{A}\mathbf{r}, \ldots, \mathbf{A}^{k-1}\mathbf{r}\}$$

  - In iteration k, CG finds the minimum of $\mathcal{J}$ over $\mathcal{K}_k(\mathbf{A}, \mathbf{b})$.
  - The kth iterate of CG achieves the minimum loss achievable in k iterations by any algorithm based on gradients and linear combinations. This includes GD and GD with momentum.
- Here, $\mathbf{A}$ is tridiagonal and $\mathbf{b} = (1, 0, \ldots, 0)^\top$.
  - What is $\mathcal{K}_k(\mathbf{A}, \mathbf{b})$?
  - How do you think CG will behave?

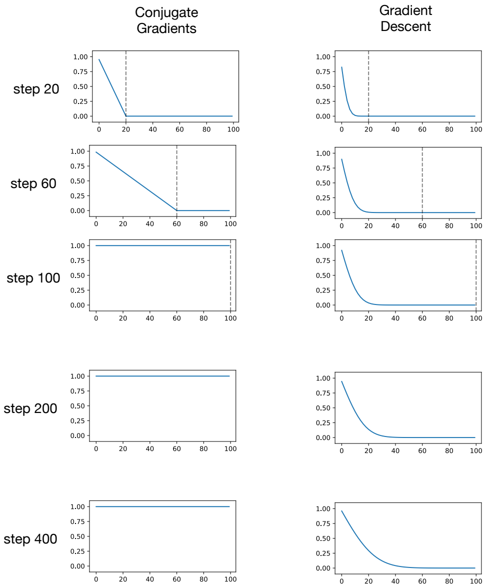
Accelerated Convergence

- Here's the behavior of CG for $N = 100$.
- **Information diffuses slowly:**
  - The Krylov subspace $\mathcal{K}_k$ is spanned by the first k coordinate vectors.
  - I.e., CG finds the optimal solution subject to all coordinates after k being 0.
  - The optimal solution interpolates linearly from 1 to 0, and achieves a loss of $1/(k+1)$
  - Subject to this constraint, the minimum achievable loss in k iterations is $1/k$
- Dashed line = speed of light.

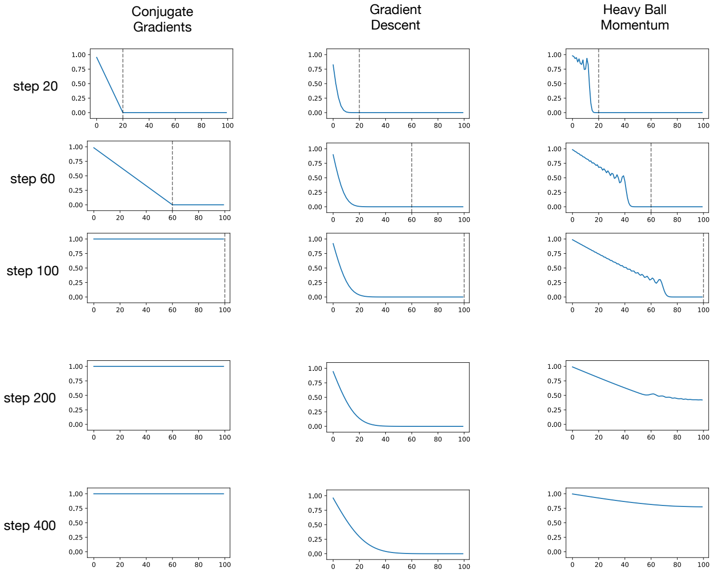


Accelerated Convergence

- Here is gradient descent with $\alpha = 0.5$ (somewhat close to optimal).
- Intuition: the gradient descent update is somewhat like diffusion. So the information travels like $\sqrt{k}$ instead of like k.
- Next slide: heavy ball momentum with $\alpha = 0.5$ and $\beta = 0.97$ (somewhat close to optimal).



Accelerated Convergence



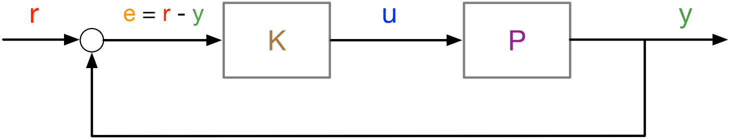
Accelerated Convergence

- HB momentum achieves the optimal convergence rate for quadratics (as we showed earlier in lecture)
- But HB doesn't achieve this convergence rate for general strongly convex functions!
  - See Lessard et al., "Analysis and design of optimization algorithms via integral quadratic constraints"
  - This paper uses techniques from control theory to automatically analyze convergence rates of first-order optimizers (including GD, HB, NAG) by solving certain semidefinite programs
  - Among many interesting contributions, they exhibit a convex function for which HB fails to achieve the optimal convergence rate
- NAG achieves the optimal convergence rate for strongly convex functions
  - Nesterov's proof is very involved, and I haven't yet seen an explanation I could cover in the scope of this lecture
- Conjugate gradient (which is exactly optimal for quadratics) often behaves a lot like HB momentum for ill-conditioned quadratics

A Controls Perspective

A Controls Perspective

- Control theory provides a powerful way to understand first-order optimization methods, including gradient descent and the various forms of momentum.
- Disclaimer: I have no formal controls background. My knowledge is limited to what Guodong and Jenny have explained to me. Errors are my own.
- In the basic control setup, one would like to choose a control signal u as input to a process P such that P's output y matches a reference signal r as closely as possible. I.e., we'd like to make the error e as small as possible.
- One does this by designing a controller K, which is typically a linear time invariant (LTI) system.

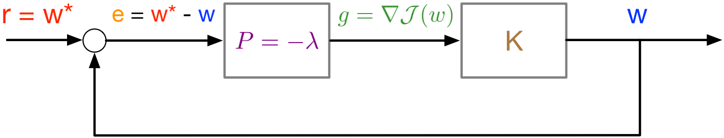


A Controls Perspective

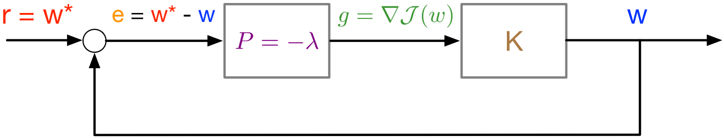
- Suppose we are trying to minimize a scalar convex quadratic:

$$\mathcal{J}(w) = \frac{\lambda}{2}(w - w^*)^2$$

- The control signal is the parameter w, and the reference signal is the optimum w^*. While w^* is fixed, for analysis it's useful to treat it as a time-varying signal.
- The process computes the gradient of the cost, which in this case is $g = \nabla \mathcal{J}(w) = \lambda(w - w^*)$.
- The controller is the optimization algorithm. It takes in the gradient, and implicitly also has a memory of past values of w.



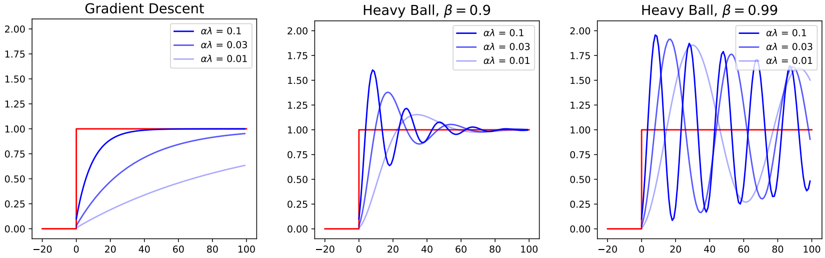
A Controls Perspective



- Since the objective is quadratic, the process is linear time invariant (LTI).
- For the controller, algorithms like GD, heavy ball momentum, and NAG are all LTI.
- Therefore, the entire system is linear, i.e. the trajectory of the optimization variable w is a linear function of the reference signal $w*$.
- It can be characterized in terms of things like the transfer function, step response, impulse response, etc. See last week's tutorial for how to do this using the z-transform.

A Controls Perspective

- The system's step response determines how it changes in response to a sudden change in the reference signal. This tells us about the convergence for a deterministic cost function.
- The idea: $w^* = 0$ for all times in the past, so the system has "settled in" to the initial value of 0. Then we begin optimizing with $w^* = 1$.
- Step responses of GD and heavy ball with various values of the curvature λ and learning rate α:



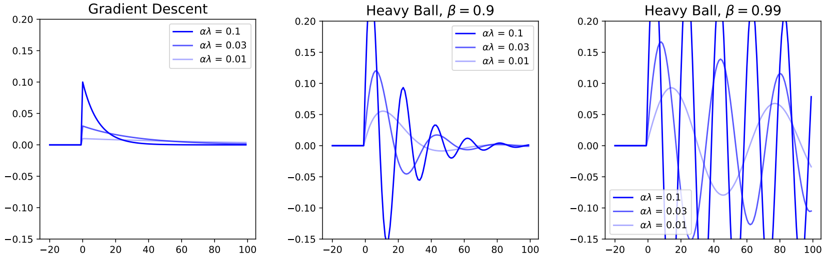
A Controls Perspective

- The impulse response h of the system is the response to an impulse (delta function).
- The parameter trajectory is obtained by convolving the reference signal with the impulse response, i.e.

$$w = w^* * h$$

$$w[t] = \sum_{\tau=0}^{\infty} h[\tau] w^*[t - \tau]$$

- Impulse responses of various optimizers:



A Controls Perspective

- The impulse function helps us understand the noise sensitivity of the optimizer. Suppose the reference signal is corrupted with i.i.d. noise ε:

$$\hat{w}^*[t] = w^*[t] + \varepsilon[t] \qquad \varepsilon[t] \sim \mathcal{N}(0, \sigma^2) \text{ for } t \geq 0.$$

- Analyzing the contribution of the noise:

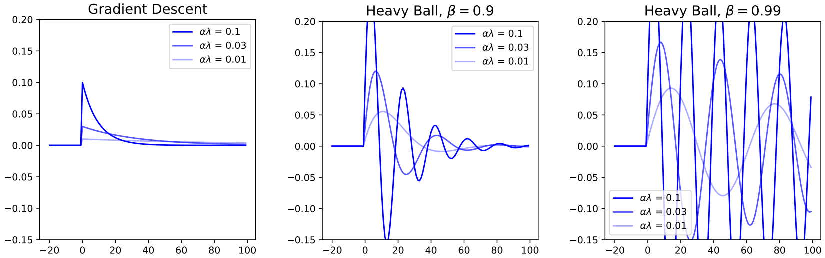
$$\begin{aligned} w[t] &= \sum_{\tau=0}^{t} h[\tau]\hat{w}^*[t-\tau] \\ &= \underbrace{\sum_{\tau=0}^{t} h[\tau]w^*[t-\tau]}_{\text{noiseless trajectory}} + \underbrace{\sum_{\tau=0}^{t} h[\tau]\varepsilon[t-\tau]}_{\text{noise term}} \end{aligned}$$

- Evaluating the noise senitivity using the basic identities of variance:

$$\text{variance of noise term} = \sigma^2 \sum_{\tau=0}^{t} h[\tau]^2.$$

- Therefore, the noise sensitivity depends on the L^2 norm of h.
- Note that h has to integrate to 1 in order for the optimizer to converge in expectation. Therefore, to minimize noise, we'd like h to be as flat as possible.

- Impulse responses of various optimizers:



- Unfortunately, the larger impulse response makes HB momentum more sensitive to noise, compared with GD.

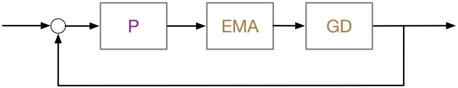
A Controls Perspective

- HB momentum can also be understood as GD with an exponential moving average of the gradients.
- Recall iterate averaging (Lecture 7): exponential moving average of the parameters:

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \alpha \nabla \mathcal{J}(\mathbf{w}^{(k)})$$
$$\tilde{\mathbf{w}}^{(k)} = \mu \tilde{\mathbf{w}}^{(k-1)} + (1-\mu)\mathbf{w}^{(k)}$$

- These algorithms essentially differ based on whether the EMA is inside or outside the optimization loop:

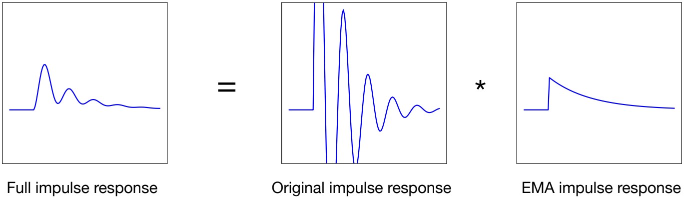
Heavy Ball Momentum



GD with Iterate Averaging

A Controls perspective

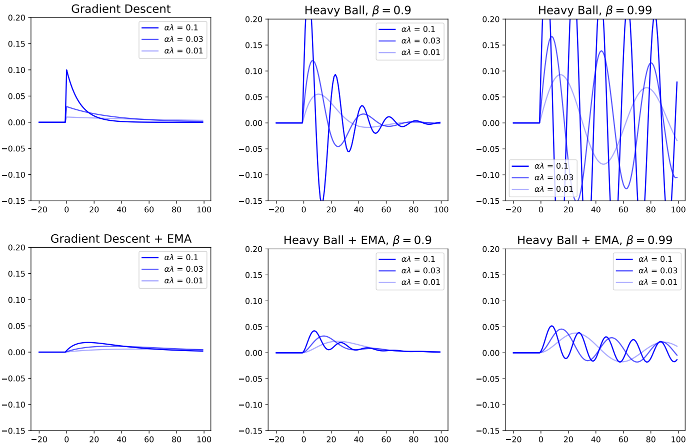
- Since iterate averaging is applied outside the optimization loop, the impulse response of the entire system is convolved with the impulse response of the EMA.



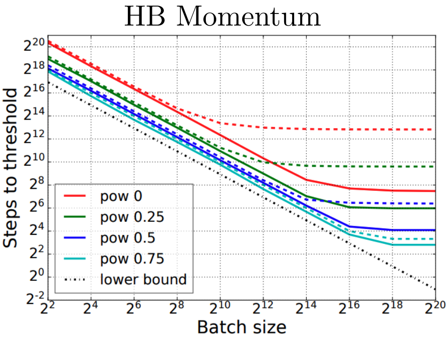
- This tends to flatten the impulse response, which reduces the noise sensitivity.

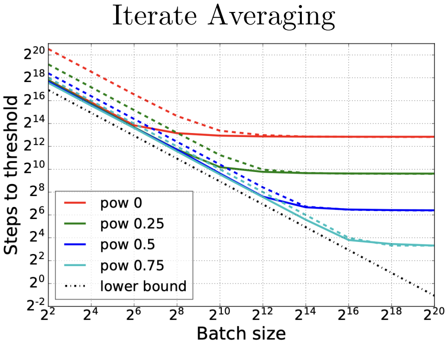
A Controls Perspective

- Iterate averaging helps flatten the impulse response, reducing noise sensitivity.



Recall from Lecture 7: HB momentum and iterate averaging, despite their superficial similarity, have very different benefits.





A Controls Perspective

- Everything I just showed you can be derived analytically using frequency domain analysis, as discussed in last week's tutorial.
- While the analytical derivations only apply to the simplified setting of quadratic objectives and i.i.d. noise, control theory has developed powerful techniques for dealing with nonlinear systems, non-i.i.d. noise, unknown dynamics, etc.
- Hu and Lessard, 2017, "Control interpretations for first-order optimization methods"
  - Interprets first-order optimization algorithms through the lens of classical control theory, shows how this can be used to design optimizers.
- Lessard et al., 2016, "Analysis and design of optimization algorithms via integral quadratic constraints"
  - Automatically analyzes the convergence rates of first-order optimizers on convex functions using stability analysis techniques from control theory.