Why Natural Gradient for General Optimization?

A brief tutorial on gradient based methods

Camille Besse

Département d'Informatique et de Génie Logiciel
besse@damas.ift.ulaval.ca
http://damas.ift.ulaval.ca

25 Septembre 2009
Practical Use of the Gradient

1. Study of force fields: Mechanics, Thermodynamics, ...
2. Study of dynamics: Differential equations, steady states, ...
3. Optimization: Gradient descent
Practical Use of the Gradient

1. Study of force fields: Mechanics, Thermodynamics, ...
2. Study of dynamics: Differential equations, steady states, ...
3. Optimization: Gradient descent
Practical Use of the Gradient

1. Study of force fields: Mechanics, Thermodynamics, ...
2. Study of dynamics: Differential equations, steady states, ...
3. **Optimization**: Gradient descent
   - For cost function minimization
   - For Policy optimization
Various Techniques for Gradient Descent

1. Randomized Search
2. Classic Gradient
3. Stochastic gradient
4. Natural gradient
Simple Notations

- Let $f(x_1, \ldots, x_n)$ be a function to optimize supposed defined and differentiable on $x_i$ domains

- Let $\nabla f = \left( \begin{array}{c} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{array} \right)$ be called the gradient of $F$
Notations throughout the Presentation

More Complex Notations

- Let $F : \begin{pmatrix} x_1 \\ ... \\ x_n \end{pmatrix} \mapsto \begin{pmatrix} f_1(x_1, ..., x_n) \\ ... \\ f_m(x_1, ..., x_n) \end{pmatrix}$ be a vectorial function

- Let $J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$ be the Jacobian matrix of $F$

- Let $G$ denotes a metric tensor defined by $G = J^\top J$. 

Natural Gradient Camille Besse 5 / 23
Example: From Cartesian to Polar Coordinates

\[ \begin{align*}
  x &= r \cos \theta \\
  y &= r \sin \theta
\end{align*} \]

- Let \( J = \begin{bmatrix}
  \frac{\partial (r \cos \theta)}{\partial r} & \frac{\partial (r \cos \theta)}{\partial \theta} \\
  \frac{\partial (r \sin \theta)}{\partial r} & \frac{\partial (r \sin \theta)}{\partial \theta}
\end{bmatrix} = \begin{bmatrix}
  \cos \theta & -r \sin \theta \\
  \sin \theta & r \cos \theta
\end{bmatrix} \]

- \( G_x = J^\top J = \begin{bmatrix} 1 & 0 \\ 0 & r_x^2 \end{bmatrix} \)

**Intuitions**

\( G \) is a similar to a *Kernel*. It is used to compute a *normalized dot product* in a *curve space*. It depends on the place where the dot product is made since the space is not necessarily orthonormal.
Randomized Gradient

**Assumptions**
- $f$ not necessarily differentiable
- $\nabla f$ not computable

**Algorithm**

Choose randomly $x_0$

**while** $||f(x_{n+1}) - f(x_n)|| > \varepsilon$ **do**

Choose a decreasing $\gamma_n$ and a random $x_{n+1}$

**if** $f(x_{n+1}) < f(x_n)$ **then**

$x_n \leftarrow x_n + 1$

**else**

$x_n \leftarrow x_n + 1$ with decreasing probability $\tau$

**end if**

**end while**

Do *many many many* random restarts

**Return** the lowest couple $x_n, f(x_n)$ found.
Pros & Cons of Randomized Search

Pros

✓ Needs only to know the function to optimize $f$

Cons

✗ May be long to converge
✗ Choice of $\text{Pr}(x_{n+1}|x_{n+1})$ really important
Classic Gradient Descent

Assumptions
- $f$ locally differentiable
- $\nabla f$ computable
- $\nabla^2 f$ computable for the choice of the best gradient step
- Search space (Parameter space) is isotropic

Algorithm
Choose randomly $x_0$
while $||f(x_{n+1}) - f(x_n)|| > \varepsilon$ do
    Choose a decreasing $\gamma_n$ (generally $\frac{1}{n}$)
    Compute $x_{n+1} = x_n - \gamma_n \nabla f(x_n)$
end while
Do some random restarts
Return the lowest couple $x_n$, $f(x_n)$ found.
Pros & Cons of Gradient Descent

Pros

✓ Converges quite fast
✓ Very efficient in Euclidian spaces

Cons

✗ Choice of $\gamma$ very important
✗ Needs $\nabla f$ and eventually $\nabla^2 f$
Stochastic Gradient Descent

Assumptions

- $f$ not necessarily differentiable
- $\nabla f$ not computable
- $f$ not necessarily computable but samples available

Algorithm

Choose randomly $x_0, x'_0$

while $\|f(x_{n+1}) - f(x_n)\| > \varepsilon$ do
  - Choose a decreasing $\gamma_n$ (generally $\frac{1}{n}$)
  - Estimate $\tilde{\nabla} f(x_n)$ using $f(x'_n)$
  - Compute $x_{n+1} = x_n - \gamma_n \nabla f(x_n)$

end while

Do many random restarts

Return the lowest couple $x_n, f(x_n)$ found.
Pros & Cons of Stochastic Gradient Descent

Pros

✓ Needs only samples of $f$

Cons

✗ Converges eventually with many samples
✗ If $f$ available, very long to converge, depending on $\gamma$
Assumptions

- $G$ is computable
- $\nabla f$ computable

Algorithm

Compute $G_x$
Choose randomly $x_0$

```
while $\|f(x_{n+1}) - f(x_n)\| > \varepsilon$ do
    Choose a decreasing $\gamma_n$ (generally $\frac{1}{n}$)
    Compute $x_{n+1} = x_n - \gamma_n G^{-1}_{x_n} \nabla f(x_n)$
end while
```

Do few random restarts
Return the lowest couple $x_n, f(x_n)$ found.
### Pros & Cons of Natural Gradient Descent

**Pros**

- Converges very fast
- Very efficient in *any* spaces

**Cons**

- Needs to compute $G$
- Needs $\nabla f$