Non Linear Optimization: Applications

Thilanka Munasinghe [1]

[1] Department of Mathematics
West Virginia University

February 24, 2015

Outline

Outline

Outline

Volatility in Finance

Volatility in Finance

Volatility

Volatility in Finance

Volatility

Volatility is a term used to describe how much the security prices, market indices, interest rates, etc., move up and down around their mean.

Volatility in Finance

Volatility

Volatility is a term used to describe how much the security prices, market indices, interest rates, etc., move up and down around their mean.

Volatility is measured by the standard deviation of the random variable that represents the financial quantity we are interested in.

Volatility in Finance

Volatility

Volatility is a term used to describe how much the security prices, market indices, interest rates, etc., move up and down around their mean.

Volatility is measured by the standard deviation of the random variable that represents the financial quantity we are interested in.

Most investors prefer low volatility to high volatility and therefore expect to be rewarded with higher long-term returns for holding higher volatility securities.

Volatility in Finance

Volatility in Finance

Volatility estimations

Volatility in Finance

Volatility estimations

Many financial computations require volatility estimates.

Volatility in Finance

Volatility estimations

Many financial computations require volatility estimates.

Most volatility estimation techniques can be classified as either a historical or an implied method.

Volatility in Finance

Volatility estimations

Many financial computations require volatility estimates.

Most volatility estimation techniques can be classified as either a historical or an implied method.

One may use historical time series to infer patterns and estimate the volatility using a statistical technique.

Volatility in Finance

Volatility estimations

Many financial computations require volatility estimates.

Most volatility estimation techniques can be classified as either a historical or an implied method.

One may use historical time series to infer patterns and estimate the volatility using a statistical technique.

An alternative would be to consider the known prices of related securities such as options that may reveal the market sentiment on the volatility of the security in question.

Volatility in Finance

Volatility estimations

Many financial computations require volatility estimates.

Most volatility estimation techniques can be classified as either a historical or an implied method.

One may use historical time series to infer patterns and estimate the volatility using a statistical technique.

An alternative would be to consider the known prices of related securities such as options that may reveal the market sentiment on the volatility of the security in question.

GARCH models exemplify the first approach, while the implied volatilities calculated from the Black, Scholes and Merton (BSM) formulas are the best known examples of the second approach.

ARCH and GARCH models

ARCH and GARCH models

The basics

ARCH and GARCH models

The basics

Let **Y** be a stochastic process indexed by natural numbers.

ARCH and GARCH models

The basics

Let $\mathbf{Y}$ be a stochastic process indexed by natural numbers.

Its value at time t, $\mathbf{Y}_t$ is an n-dimensional vector of random variables.

ARCH and GARCH models

The basics

Let **Y** be a stochastic process indexed by natural numbers.

Its value at time t, $\mathbf{Y}_t$ is an n-dimensional vector of random variables.

The behavior of these random variables is modeled as

ARCH and GARCH models

The basics

Let **Y** be a stochastic process indexed by natural numbers.

Its value at time t, $\mathbf{Y}_t$ is an n-dimensional vector of random variables.

The behavior of these random variables is modeled as

$$\mathbf{Y}_t = \sum_{i=1}^{m} \Phi_i \cdot \mathbf{Y}_{t-i} + \epsilon_t.$$

ARCH and GARCH models

The basics

Let $\mathbf{Y}$ be a stochastic process indexed by natural numbers.

Its value at time t, $\mathbf{Y}_t$ is an n-dimensional vector of random variables.

The behavior of these random variables is modeled as

$$\mathbf{Y}_t = \sum_{i=1}^{m} \Phi_i \cdot \mathbf{Y}_{t-i} + \epsilon_t.$$

Here m is a positive integer representing the number of periods we look back in our model and ϵ_t satisfies:

ARCH and GARCH models

The basics

Let **Y** be a stochastic process indexed by natural numbers.

Its value at time t, $\mathbf{Y}_t$ is an n-dimensional vector of random variables.

The behavior of these random variables is modeled as

$$\mathbf{Y}_t = \sum_{i=1}^{m} \Phi_i \cdot \mathbf{Y}_{t-i} + \epsilon_t.$$

Here m is a positive integer representing the number of periods we look back in our model and ϵ_t satisfies:

$$\mathbf{E}[\epsilon_t | \epsilon_1, ..., \epsilon_{t-1}] = 0.$$

ARCH and GARCH models

ARCH and GARCH models

The model for the univariate case

ARCH and GARCH models

The model for the univariate case

We set:

$$h_t = \mathbf{E}[\epsilon_t^2 | \epsilon_1, ..., \epsilon_{t-1}].$$

ARCH and GARCH models

The model for the univariate case

We set:

$$h_t = \mathbf{E}[\epsilon_t^2 | \epsilon_1, ..., \epsilon_{t-1}].$$

Then we model the conditional time dependence as follows:

ARCH and GARCH models

The model for the univariate case

We set:

$$h_t = \mathbf{E}[\epsilon_t^2 | \epsilon_1, ..., \epsilon_{t-1}].$$

Then we model the conditional time dependence as follows:

$$h_t = c + \sum_{i=1}^{q} \alpha_i \cdot \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \cdot h_{t-j}.$$

ARCH and GARCH models

The model for the univariate case

We set:

$$h_t = \mathbf{E}[\epsilon_t^2 | \epsilon_1, ..., \epsilon_{t-1}].$$

Then we model the conditional time dependence as follows:

$$h_t = c + \sum_{i=1}^{q} \alpha_i \cdot \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \cdot h_{t-j}.$$

This model is called GARCH(p,q).

ARCH and GARCH models

The model for the univariate case

We set:

$$h_t = \mathbf{E}[\epsilon_t^2 | \epsilon_1, ..., \epsilon_{t-1}].$$

Then we model the conditional time dependence as follows:

$$h_t = c + \sum_{i=1}^{q} \alpha_i \cdot \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \cdot h_{t-j}.$$

This model is called GARCH(p,q).

ARCH model corresponds to choosing $p = 0$.

ARCH and GARCH models

ARCH and GARCH models

The optimization problem in the univariate case

ARCH and GARCH models

The optimization problem in the univariate case

After choosing p and q, the objective is to determine the optimal parameters Φ_i, α_i, β_j.

ARCH and GARCH models

The optimization problem in the univariate case

After choosing p and q, the objective is to determine the optimal parameters Φ_i, α_i, β_j.

Usually, this is achieved via maximum likelihood estimation.

ARCH and GARCH models

The optimization problem in the univariate case

After choosing p and q, the objective is to determine the optimal parameters Φ_i, α_i, β_j.

Usually, this is achieved via maximum likelihood estimation.

If we assume that $\mathbf{Y}_t$ has a normal distribution conditional on the historical observations, the log-likelihood function can be written as follows:

ARCH and GARCH models

The optimization problem in the univariate case

After choosing p and q, the objective is to determine the optimal parameters Φ_i, α_i, β_j.

Usually, this is achieved via maximum likelihood estimation.

If we assume that $\mathbf{Y}_t$ has a normal distribution conditional on the historical observations, the log-likelihood function can be written as follows:

$$-\frac{T}{2} \cdot \ln(2 \cdot \pi) - \frac{1}{2} \cdot \sum_{i=1}^{T} \ln h_t - \frac{1}{2} \cdot \sum_{i=1}^{T} \frac{\epsilon_t^2}{h_t}.$$

ARCH and GARCH models

ARCH and GARCH models

The optimization problem in the univariate case

ARCH and GARCH models

The optimization problem in the univariate case

$$\max\left(-\frac{T}{2}\cdot\ln(2\cdot\pi)-\frac{1}{2}\cdot\sum_{i=1}^{T}\ln h_t-\frac{1}{2}\cdot\sum_{i=1}^{T}\frac{\epsilon_t^2}{h_t}\right)$$

ARCH and GARCH models

The optimization problem in the univariate case

$$\max(-\frac{T}{2}\cdot\ln(2\cdot\pi)-\frac{1}{2}\cdot\sum_{i=1}^{T}\ln h_t-\frac{1}{2}\cdot\sum_{i=1}^{T}\frac{\epsilon_t^2}{h_t})$$

$$\mathbf{Y}_t=\sum_{i=1}^{m}\Phi_i\cdot\mathbf{Y}_{t-i}+\epsilon_t$$

ARCH and GARCH models

The optimization problem in the univariate case

$$\max\left(-\frac{T}{2}\cdot\ln(2\cdot\pi)-\frac{1}{2}\cdot\sum_{i=1}^{T}\ln h_t-\frac{1}{2}\cdot\sum_{i=1}^{T}\frac{\epsilon_t^2}{h_t}\right)$$

$$\mathbf{Y}_t=\sum_{i=1}^{m}\Phi_i\cdot\mathbf{Y}_{t-i}+\epsilon_t$$

$$\mathbf{E}[\epsilon_t|\epsilon_1,...,\epsilon_{t-1}]=0$$

ARCH and GARCH models

The optimization problem in the univariate case

$$\max\left(-\frac{T}{2}\cdot\ln(2\cdot\pi)-\frac{1}{2}\cdot\sum_{i=1}^{T}\ln h_t-\frac{1}{2}\cdot\sum_{i=1}^{T}\frac{\epsilon_t^2}{h_t}\right)$$

$$\mathbf{Y}_t=\sum_{i=1}^{m}\Phi_i\cdot\mathbf{Y}_{t-i}+\epsilon_t$$

$$\mathbf{E}[\epsilon_t|\epsilon_1,...,\epsilon_{t-1}]=0$$

$$h_t=\mathbf{E}[\epsilon_t^2|\epsilon_1,...,\epsilon_{t-1}]\geq 0,\ \text{for all } t$$

ARCH and GARCH models

ARCH and GARCH models

Stationarity properties

ARCH and GARCH models

Stationarity properties

- An important issue in GARCH parameter estimations is the stationarity properties of the resulting model.

ARCH and GARCH models

Stationarity properties

- An important issue in GARCH parameter estimations is the stationarity properties of the resulting model.
- It is unclear whether one can assume that the model parameters for financial time series are stationary over the time.

ARCH and GARCH models

Stationarity properties

- An important issue in GARCH parameter estimations is the stationarity properties of the resulting model.
- It is unclear whether one can assume that the model parameters for financial time series are stationary over the time.
- However, the forecasting and estimation is easier on stationary models.

ARCH and GARCH models

Stationarity properties

- An important issue in GARCH parameter estimations is the stationarity properties of the resulting model.
- It is unclear whether one can assume that the model parameters for financial time series are stationary over the time.
- However, the forecasting and estimation is easier on stationary models.

Theorem

If α_i's, β_j's and the scalar c are strictly positive and

$$\sum_{i=1}^{q} \alpha_i + \sum_{j=1}^{p} \beta_j < 1,$$

ARCH and GARCH models

Stationarity properties

- An important issue in GARCH parameter estimations is the stationarity properties of the resulting model.
- It is unclear whether one can assume that the model parameters for financial time series are stationary over the time.
- However, the forecasting and estimation is easier on stationary models.

Theorem

If α_i's, β_j's and the scalar c are strictly positive and

$$\sum_{i=1}^{q} \alpha_i + \sum_{j=1}^{p} \beta_j < 1,$$

then the univariate GARCH model is stationary.

Black-Scholes-Merton (BSM) equations

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

where

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

where

- S_t- the underlying security price at time t,

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

where

- S_t- the underlying security price at time t,
- μ- drift,

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

where

- S_t- the underlying security price at time t,
- μ- drift,
- σ- the constant volatility,

Black-Scholes-Merton (BSM) equations

BSM for European pricing option

$$\frac{dS_t}{S_t} = \mu \cdot dt + \sigma \cdot dW_t,$$

where

- S_t- the underlying security price at time t,
- μ- drift,
- σ- the constant volatility,
- W_t- a random variable.

Black-Scholes-Merton (BSM) equations

Black-Scholes-Merton (BSM) equations

The solutions

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

where

$$d_1 = \frac{\log(S_0/K) + (r + \sigma^2/2) \cdot T)}{\sigma \cdot \sqrt{T}},$$

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

where

$$d_1 = \frac{\log(S_0/K) + (r + \sigma^2/2) \cdot T)}{\sigma \cdot \sqrt{T}},$$
$$d_2 = d_1 - \sigma \cdot \sqrt{T},$$

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

where

$$d_1 = \frac{\log(S_0/K) + (r + \sigma^2/2) \cdot T)}{\sigma \cdot \sqrt{T}},$$
$$d_2 = d_1 - \sigma \cdot \sqrt{T},$$

- $\Phi()$-cumulative distribution function for the standard normal distribution,

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

where

$$d_1 = \frac{\log(S_0/K) + (r + \sigma^2/2) \cdot T)}{\sigma \cdot \sqrt{T}},$$
$$d_2 = d_1 - \sigma \cdot \sqrt{T},$$

- $\Phi()$-cumulative distribution function for the standard normal distribution,
- r-continuously compounded risk-free interest rate (a constant available in US markets),

Black-Scholes-Merton (BSM) equations

The solutions

$$C(K,T) = S_0 \cdot \Phi(d_1) - K \cdot e^{-r \cdot T} \cdot \Phi(d_2),$$
$$P(K,T) = K \cdot e^{-r \cdot T} \cdot \Phi(-d_2) - S_0 \cdot \Phi(-d_1),$$

where

$$d_1 = \frac{\log(S_0/K) + (r + \sigma^2/2) \cdot T)}{\sigma \cdot \sqrt{T}},$$
$$d_2 = d_1 - \sigma \cdot \sqrt{T},$$

- $\Phi()$-cumulative distribution function for the standard normal distribution,
- r-continuously compounded risk-free interest rate (a constant available in US markets),
- σ- the constant volatility.

Black-Scholes-Merton (BSM) equations

Black-Scholes-Merton (BSM) equations

Implied volatility

Black-Scholes-Merton (BSM) equations

Implied volatility

In order to determine the price, we just need the value or a close approximation to σ.

Black-Scholes-Merton (BSM) equations

Implied volatility

In order to determine the price, we just need the value or a close approximation to σ.

Conversely, given the market price for a particular European call or put, one can determine the volatility (implied by the price), called implied volatility, by solving these equations with the unknown σ.

Linear Vs Non Linear

Definition of Linear and Non Linear

$$f(x) = x_1 + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is linear in } x = [x_1, x_2, x_3]^T$$

Linear Vs Non Linear

Definition of Linear and Non Linear

$$f(x) = x_1 + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is linear in } x = [x_1, x_2, x_3]^T$$

$$f(x) = x_1 \cdot x_2 + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is Non Linear in } x$$

Linear Vs Non Linear

Definition of Linear and Non Linear

$$f(x) = x_1 + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is linear in } x = [x_1, x_2, x_3]^T$$

$$f(x) = x_1 \cdot x_2 + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is Non Linear in } x$$

$$f(x) = cos(x_1) + 2 \cdot x_2 - (3.4) \cdot x_3 \text{ is Non Linear in } x$$

Existence and Uniqueness of the an Optimum Solution

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero
- Vector Case : at minimum, Hessian is positive definite.

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero
- Vector Case : at minimum, Hessian is positive definite.

Remark

Necessary and Sufficient conditions for a minimum for an unconstrained problem :

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero
- Vector Case : at minimum, Hessian is positive definite.

Remark

Necessary and Sufficient conditions for a minimum for an unconstrained problem : Gradient must equal to zero,

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero
- Vector Case : at minimum, Hessian is positive definite.

Remark

Necessary and Sufficient conditions for a minimum for an unconstrained problem : Gradient must equal to zero, $\bigtriangledown f(x^*) = 0$.

Existence and Uniqueness of the an Optimum Solution

Existence and uniqueness of the solution

- Usually can not guarantee that we have found the Global Optima.
- There can be multiple solutions that exist.
- For unconstrained problems, at the minimum, $\bigtriangledown f(x^*) = 0$
- Calculas : at minimum, the Second Derivative is greater than zero
- Vector Case : at minimum, Hessian is positive definite.

Remark

Necessary and Sufficient conditions for a minimum for an unconstrained problem : Gradient must equal to zero, $\bigtriangledown f(x^*) = 0$*. Hessian must be positive definite.*

Line Search Methods

The Formulation of the Line Search Method

Consider an unconstrained optimization problem,

Line Search Methods

The Formulation of the Line Search Method

Consider an unconstrained optimization problem,

$$\min f(x)$$

Line Search Methods

The Formulation of the Line Search Method

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R$$

Line Search Methods

The Formulation of the Line Search Method

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R$$

Assume the function $f(x)$ is smooth and continuous.

Line Search Methods

The Formulation of the Line Search Method

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R$$

Assume the function $f(x)$ is smooth and continuous.

The objective is to find a minimum of $f(x)$.

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimal Point

Goal is to find the "optimal point" x^*

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimal Point

Goal is to find the "optimal point" x^*

Iteration equation / Iteration Scheme

$$x_{k+1} = x_k + \alpha_k \cdot d_k$$

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimal Point

Goal is to find the "optimal point" x^*

Iteration equation / Iteration Scheme

$$x_{k+1} = x_k + \alpha_k \cdot d_k$$

d_k = "search direction".

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimal Point

Goal is to find the "optimal point" x^*

Iteration equation / Iteration Scheme

$$x_{k+1} = x_k + \alpha_k \cdot d_k$$

d_k = "search direction".

α_k = "step-length"

Optimization Algorithm

Initial point

Optimization Algorithm starts by an initial point x_0, and performs series of iterations.

Optimal Point

Goal is to find the "optimal point" x^*

Iteration equation / Iteration Scheme

$$x_{k+1} = x_k + \alpha_k \cdot d_k$$

d_k = "search direction".

α_k = "step-length"

step-length, α_k determines how far to go on the "search direction", d_k

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,* x_0.

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point, x_0.*
- *Find the decent "search direction",*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,* x_0*.*
- *Find the decent "search direction",*d_k*.*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point, x_0.*
- *Find the decent "search direction", d_k.*
- *Determine the "step-size",*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,* x_0.
- *Find the decent "search direction",* d_k.
- *Determine the "step-size",* α_k.

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point, x_0.*
- *Find the decent "search direction", d_k.*
- *Determine the "step-size", α_k.*
- *Check the iteration criteria.*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,* x_0.
- *Find the decent "search direction",* d_k.
- *Determine the "step-size",* α_k.
- *Check the iteration criteria.*
- *Check the stopping conditions.*

Constraints

Definition

The function value of the new point, $f(x_{k+1})$ should be less than or equal to the previous point function value, $f(x_k)$.

Constraint

$$f(x_{k+1}) \leq f(x_k)$$
$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k)$$

Remark

- *Optimization Algorithm starts with an initial point,* x_0.
- *Find the decent "search direction",* d_k.
- *Determine the "step-size",* α_k.
- *Check the iteration criteria.*
- *Check the stopping conditions.*
- *Output the Optimal point,* x^*

Newton's Method

Definition

Newton's method

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is $f'(x)$, Calculation begins with a first guessing of the initial point

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is $f'(x)$, Calculation begins with a first guessing of the initial point x_0 for a root of the function $f(x)$.

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is $f'(x)$, Calculation begins with a first guessing of the initial point x_0 for a root of the function $f(x)$. A new, considerably a better approximation point,

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is $f'(x)$, Calculation begins with a first guessing of the initial point x_0 for a root of the function $f(x)$. A new, considerably a better approximation point, x_1
which obtained by,

Newton's Method

Definition

Newton's method (also known as the Newton-Raphson method)

Uses to finding successively better approximations to the roots (or zeroes) of a real-valued function defined on the interval [a, b]

$$f(x) = 0$$

Where $f(x)$ is continuous and differentiable.

Definition

Given a function $f(x)$ is defined over the $x \in \mathbb{R}$, and its first derivative is $f'(x)$, Calculation begins with a first guessing of the initial point x_0 for a root of the function $f(x)$. A new, considerably a better approximation point, x_1
which obtained by,

$$x_1 = x_0 + \frac{f(x_0)}{f'(x_0)}$$

Newton's Method

Iterative Scheme

As the iterative process repeats,

Newton's Method

Iterative Scheme

As the iterative process repeats, current approximation x_n is used to derive the formula for a new, better approximation, x_{n+1}

Newton's Method

Iterative Scheme

As the iterative process repeats, current approximation x_n is used to derive the formula for a new, better approximation, x_{n+1}

$$x_{n+1} = x_n + \frac{f(x_n)}{f'(x_n)}$$

Newton's Method

Tangent Line

Suppose we have some current approximation x_n.

Newton's Method

Tangent Line

Suppose we have some current approximation x_n. Then we can derive the formula for a better approximation, x_{n+1}. The equation of the tangent line to the curve $y = f(x)$ at the point $x = x_n$ is,

Newton's Method

Tangent Line

Suppose we have some current approximation x_n. Then we can derive the formula for a better approximation, x_{n+1}. The equation of the tangent line to the curve $y = f(x)$ at the point $x = x_n$ is,

$$y = f'(x_n)(x_{n+1} - x_n) + f(x_n)$$

Newton's Method

Tangent Line

Suppose we have some current approximation x_n. Then we can derive the formula for a better approximation, x_{n+1}. The equation of the tangent line to the curve $y = f(x)$ at the point $x = x_n$ is,

$$y = f'(x_n)(x_{n+1} - x_n) + f(x_n)$$

Definition

The x-intercept of this line (the value of x such that $y = 0$) is then used as the next approximation to the root, x_{n+1}. In other words, setting $y = 0$ and $x = x_{n+1}$ gives

Newton's Method

Tangent Line

Suppose we have some current approximation x_n. Then we can derive the formula for a better approximation, x_{n+1}. The equation of the tangent line to the curve $y = f(x)$ at the point $x = x_n$ is,

$$y = f'(x_n)(x_{n+1} - x_n) + f(x_n)$$

Definition

The x-intercept of this line (the value of x such that $y = 0$) is then used as the next approximation to the root, x_{n+1}. In other words, setting $y = 0$ and $x = x_{n+1}$ gives

$$0 = f'(x_n)(x_{n+1} - x_n) + f(x_n)$$

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point,

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

$$f(\bar{x} + d) \approx f(\bar{x}) + \triangledown f(\bar{x})^T d$$

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

$$f(\bar{x} + d) \approx f(\bar{x}) + \bigtriangledown f(\bar{x})^T d$$

if $\|d\|$ is small, notice that the approximation above is good.

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

$$f(\bar{x} + d) \approx f(\bar{x}) + \nabla f(\bar{x})^T d$$

if $\|d\|$ is small, notice that the approximation above is good.

We choose the value of d so that the inner product,

Steepest Descent Method

Steepest Descent Algorithm for Unconstrained Optimization

Consider an unconstrained optimization problem,

$$\min f(x)$$
$$x \in R^n$$

Assume the function $f(x)$ is smooth and continuous and differentiable.
If, $x = \bar{x}$ is a given point, $f(x)$ can be approximated by its linear expansion

$$f(\bar{x} + d) \approx f(\bar{x}) + \bigtriangledown f(\bar{x})^T d$$

if $\|d\|$ is small, notice that the approximation above is good.

We choose the value of d so that the inner product, $\bigtriangledown f(\bar{x})^T d$ is small as possible.

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x}\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

$$\bigtriangledown f(\bar{x})^T d \geq -\| \bigtriangledown f(\bar{x})\| \|d\| = \bigtriangledown f(\bar{x})^T (\frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}) = - \bigtriangledown f(\tilde{d}).$$

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

$$\bigtriangledown f(\bar{x})^T d \geq -\| \bigtriangledown f(\bar{x})\|\|d\| = \bigtriangledown f(\bar{x})^T \left(\frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}\right) = - \bigtriangledown f(\tilde{d}).$$

Direction of the Steepest Descent

Due to the above reason,

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

$$\bigtriangledown f(\bar{x})^T d \geq -\| \bigtriangledown f(\bar{x})\| \|d\| = \bigtriangledown f(\bar{x})^T \left(\frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}\right) = - \bigtriangledown f(\tilde{d}).$$

Direction of the Steepest Descent

Due to the above reason, Steepest Descent Direction :

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

$$\bigtriangledown f(\bar{x})^T d \geq -\| \bigtriangledown f(\bar{x})\| \|d\| = \bigtriangledown f(\bar{x})^T \left(\frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}\right) = - \bigtriangledown f(\tilde{d}).$$

Direction of the Steepest Descent

Due to the above reason, Steepest Descent Direction :

$$\bar{d} = - \bigtriangledown f(\bar{x})$$

Steepest Descent Method

Gradient vector

Let's normalize d so that, $\|d\| = 1$.
Then among all directions, d with norm $\|d\| = 1$,
the direction,

$$\tilde{d} = \frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}$$

makes the smallest inner product with the gradient, $\bigtriangledown f(\bar{x})$.
This fact follows from the following inequalities:

$$\bigtriangledown f(\bar{x})^T d \geq -\| \bigtriangledown f(\bar{x})\| \|d\| = \bigtriangledown f(\bar{x})^T \left(\frac{-\bigtriangledown f(\bar{x})}{\|\bigtriangledown f(\bar{x})\|}\right) = - \bigtriangledown f(\tilde{d}).$$

Direction of the Steepest Descent

Due to the above reason, Steepest Descent Direction :

$$\bar{d} = - \bigtriangledown f(\bar{x})$$

This is called the "**direction of the steepest descent**" at point $\bar{x}$.

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Step 2. $d_k = -\bigtriangledown f(x_k)$. If $d_k = 0$, then stop.

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.

Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.

Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k

Step 4. Set $x_{k+1} \leftarrow x_k + \alpha_k \cdot d_k$, $k \leftarrow k + 1$

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

- Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$
- Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.
- Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k
- Step 4. Set $x_{k+1} \leftarrow x_k + \alpha_k \cdot d_k$, $k \leftarrow k + 1$

Note from Step 3,

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

- Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$
- Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.
- Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k
- Step 4. Set $x_{k+1} \leftarrow x_k + \alpha_k \cdot d_k$, $k \leftarrow k + 1$

Note from Step 3, the fact that $d_k = - \bigtriangledown f(x_k)$ is a descent direction,

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.

Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k

Step 4. Set $x_{k+1} \leftarrow x_k + \alpha_k \cdot d_k$, $k \leftarrow k + 1$

Note from Step 3, the fact that $d_k = - \bigtriangledown f(x_k)$ is a descent direction, which follows the condition of

Steepest Descent Algorithm

Steps of the SD Algorithm

Steps are

Step 1. Initialize x^0 and machine accuracy ε , set $k = 0$

Step 2. $d_k = - \bigtriangledown f(x_k)$. If $d_k = 0$, then stop.

Step 3. Solve $min_\alpha f(x_k + \alpha_k \cdot d_k)$ for the stepsize α_k

Step 4. Set $x_{k+1} \leftarrow x_k + \alpha_k \cdot d_k$, $k \leftarrow k + 1$

Note from Step 3, the fact that $d_k = - \bigtriangledown f(x_k)$ is a descent direction, which follows the condition of

$$f(x_k + \alpha_k \cdot d_k) \leq f(x_k) \ .$$

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by :

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by : $\bigtriangledown f(x_k) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

$$\text{Taking } x_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix},$$

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by : $\bigtriangledown f(x_k) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

$$\text{Taking } x_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \text{, we have } \bigtriangledown f(x_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Performing line search along negative gradient direction,

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by : $\bigtriangledown f(x_k) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

$$\text{Taking } x_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \text{, we have } \bigtriangledown f(x_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Performing line search along negative gradient direction,

$$min_{\alpha_0} f(x_0 - \alpha_0 \bigtriangledown f(x_o))$$

Exact minimum along line is given by $\alpha_0 = 1/3$,

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by : $\bigtriangledown f(x_k) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

$$\text{Taking } x_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \text{, we have } \bigtriangledown f(x_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Performing line search along negative gradient direction,

$$min_{\alpha_0} f(x_0 - \alpha_0 \bigtriangledown f(x_o))$$

Exact minimum along line is given by $\alpha_0 = 1/3$,
so next approximation is

Example of Steepest Descent Method

Using SD Method to minimize f(x)

Find the the first iteration value of x^* using the Steepest Descent Method :

$$\min f(x) = 0.5x_1^2 + 2.5x_2^2$$

Gradient is given by : $\bigtriangledown f(x_k) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

$$\text{Taking } x_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \text{, we have } \bigtriangledown f(x_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Performing line search along negative gradient direction,

$$min_{\alpha_0} f(x_0 - \alpha_0 \bigtriangledown f(x_o))$$

Exact minimum along line is given by $\alpha_0 = 1/3$,
so next approximation is

$$x_1 = \begin{bmatrix} 3.333 \\ -0.667 \end{bmatrix}$$

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either,

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either, $(x_2, b]$ or $[a, x_1)$, with minimum known to lie in remaining subintervals.

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either, $(x_2, b]$ or $[a, x_1)$, with minimum known to lie in remaining subintervals.

In order to repeat the process,

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either, $(x_2, b]$ or $[a, x_1)$, with minimum known to lie in remaining subintervals.

In order to repeat the process, we need to compute only one new function evaluation.

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either, $(x_2, b]$ or $[a, x_1)$, with minimum known to lie in remaining subintervals.

In order to repeat the process, we need to compute only one new function evaluation.

To reduce length of an interval by a fixed fraction at each iteration,

Golden Section Search Method

Definition

Suppose $f(x)$ is unimodal on $[a, b]$,
let x_1 and x_2 be two points within $[a, b]$,

$$\text{where } x_1 < x_2$$

Evaluating and comparing the values of $f(x_1)$ and $f(x_2)$, we can discard either, $(x_2, b]$ or $[a, x_1)$, with minimum known to lie in remaining subintervals.

In order to repeat the process, we need to compute only one new function evaluation.

To reduce length of an interval by a fixed fraction at each iteration, each new pair of points must have the same relationship with respect to new interval that previous pair had with respect to that previous interval.

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else
$b = x_2$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else
$b = x_2$
$x_2 = x_1$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else
$b = x_2$
$x_2 = x_1$
$x_1 = a + (1 - \tau)(b - a)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else
$b = x_2$
$x_2 = x_1$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$

Golden Section Search Algorithm

Consider $\tau = (\sqrt{5} - 1)/2$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
$x_2 = a + \tau(b - a)$;
$f_2 = f(x_2)$
While
$((b - a) > tol)$ Do
if $(f_1 > f_2)$ then
$a = x_1$
$x_1 = x_2$
$f_1 = f_2$
$x_2 = a + \tau(b - a)$
$f_2 = f(x_2)$
Else
$b = x_2$
$x_2 = x_1$
$x_1 = a + (1 - \tau)(b - a)$
$f_1 = f(x_1)$
End

Definition

- Another method that does not require explicit second derivatives, and does not even store approximation to the Hessian matrix is,

Definition

- Another method that does not require explicit second derivatives, and does not even store approximation to the Hessian matrix is, Conjugate Gradient (CG) method.
- CG generates sequence of conjugate search directions,

Definition

- Another method that does not require explicit second derivatives, and does not even store approximation to the Hessian matrix is, Conjugate Gradient (CG) method.
- CG generates sequence of conjugate search directions, implicitly accumulating information about Hessian matrix.
- For quadratic objective function,

Definition

- Another method that does not require explicit second derivatives, and does not even store approximation to the Hessian matrix is, Conjugate Gradient (CG) method.
- CG generates sequence of conjugate search directions, implicitly accumulating information about Hessian matrix.
- For quadratic objective function, CG is theoretically exact after at most n iterations,

Definition

- Another method that does not require explicit second derivatives, and does not even store approximation to the Hessian matrix is, Conjugate Gradient (CG) method.
- CG generates sequence of conjugate search directions, implicitly accumulating information about Hessian matrix.
- For quadratic objective function, CG is theoretically exact after at most n iterations,where n is the dimension of the problem.
- CG is effective for general unconstrained minimization.