Interest Rate Modelling and Derivative Pricing

Sebastian Schlenkrich

HU Berlin, Department of Mathematics

WS, 2018/19

Part VII

Sensitivity Calculation

Outline

Why do we need sensitivities?

Consider a (differentiable) pricing model $V = V(p)$ based on some input parameter p. Sensitivity of V w.r.t. changes in p is

$$V'(p) = \frac{dV(p)}{dp}.$$

- Hedging and risk management
- Market risk measurement
- Many more applications for accounting, regulatory reporting,...

Sensitivity calculation is a crucial function for banks and financial institutions.

Derivative pricing is based on hedging and risk replication

Recall fundamental derivative replication result

$$V(t) = V(t, X(t)) = \phi(t)^\top X(t) \text{ for all } t \in [0, T]$$

- $V(t)$ price of a contingent claim
- $\phi(t)$ permissible trading strategy
- $X(t)$ assets in our market

How do we find the trading strategy?

Consider portfolio $\pi(t) = V(t, X(t)) - \phi(t)^\top X(t)$ and apply Ito's lemma

$$d\pi(t) = \mu_\pi \cdot dt + [\nabla_X \pi(t)]^\top \cdot \sigma_X^\top dW(t)$$

From replication property follows $d\pi(t) = 0$ for all $t \in [0, T]$. Thus, in particular

$$0 = \nabla_X \pi(t) = \nabla_X V(t, X(t)) - \phi(t)$$

This gives Delta-hedge

$$\phi(t) = \nabla_X V(t, X(t))$$

Market risk calculation relies on accurate sensitivities

Consider portfolio value $\pi(t)$, time horizon Δt and returns

$$\Delta\pi(t) = \pi(t - \Delta t) - \pi(t).$$

Market risk measure Value at Risk (VaR) is the lower quantile q of distribution of portfolio returns $\Delta\pi(t)$ given a confidence level $1 - \alpha$, formally

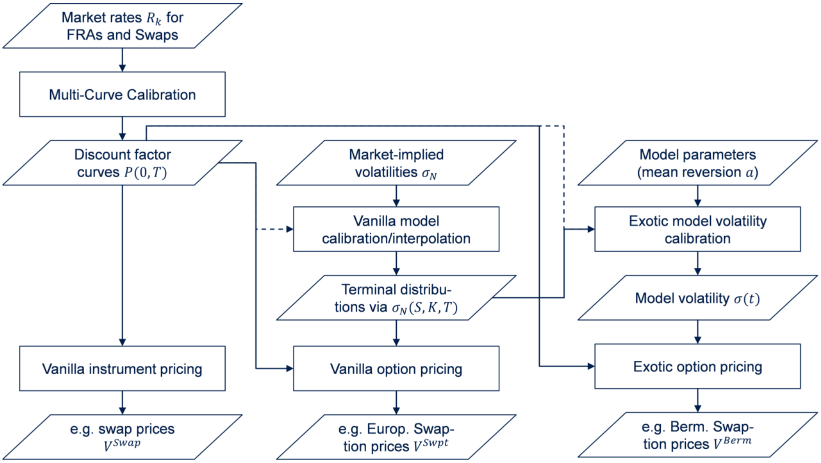
$$\text{VaR}_\alpha = \inf\left\{q \quad s.t.\, \mathbb{P}\left\{\Delta\pi(t) \le q \,|\, \pi(t)\right\} > \alpha\right\}.$$

Delta-Gamma VaR calculation method consideres $\pi(t) = \pi\left(X(t)\right)$ in terms of risk factors $X(t)$ and approximates

$$\Delta\pi \approx \left[\nabla_X \pi\left(X\right)\right]^\top \Delta X + \frac{1}{2}\Delta X^\top \left[H_X \pi\left(X\right)\right]\Delta X.$$

- VaR is calculated based on joint distribution of risk factor returns $\Delta X = X(t + \Delta t) - X(t)$ and sensitivities $\nabla_X \pi$ (gradient) and $H_X \pi$ (Hessian)
- Bank portfolio π may consist of linear instruments (e.g. swaps), Vanilla options (e.g. European swaptions) and exotic instruments (e.g. Bermudans)
- Common interest rate risk factors are FRA rates, par swap rates, ATM volatilities

Sensitivity specification needs to take into account data flow and dependencies



Market rates R_k for FRAs and Swaps
Multi-Curve Calibration
Discount factor curves $P(0,T)$
Market-implied volatilities σ_N
Model parameters (mean reversion a)
Vanilla model calibration/interpolation
Exotic model volatility calibration
Terminal distributions via $\sigma_N(S,K,T)$
Model volatility $\sigma(t)$
Vanilla instrument pricing
Vanilla option pricing
Exotic option pricing
e.g. swap prices V^{Swap}
e.g. Europ. Swaption prices V^{Swpt}
e.g. Berm. Swaption prices V^{Berm}

Depending on context, risk factors can be market parameters or model parameters.

In practice, sensitivities are scaled relative to pre-defined risk factor shifts

Scaled sensitivity ΔV becomes

$$\Delta V = \frac{dV(p)}{dp} \cdot \Delta p \approx V(p + \Delta p) - V(p)$$

Typical scaling (or risk factor shift sizes) Δp are

- $1bp$ for interest rate shifts
- $1bp$ for implied normal volatilities
- 1% for implied lognormal or shifted lognormal volatilities

Par rate Delta and Gamma are sensitivity w.r.t. changes in market rates I

Bucketed Delta and Gamma

Let $\bar{R} = [R_k]_{k=1,\dots q}$ be the list of market quotes defining the inputs of a yield curve. The bucketed par rate delta of an instrument with model price $V = V(\bar{R})$ is the vector

$$\Delta_R = 1bp \cdot \left[\frac{\partial V}{\partial R_1}, \dots, \frac{\partial V}{\partial R_q} \right].$$

Bucketed Gamma is calculated as

$$\Gamma_R = [1bp]^2 \cdot \left[\frac{\partial^2 V}{\partial R_1^2}, \dots, \frac{\partial^2 V}{\partial R_q^2} \right].$$

- For multiple projection and discounting yield curves, sensitivities are calculated for each curve individually

Par rate Delta and Gamma are sensitivity w.r.t. changes in market rates II

Parallel Delta and Gamma

Parallel Delta and Gamma represent sensitivities w.r.t. simultanous shifts of all market rates of a yield curve. With $\mathbf{1} = [1, \ldots 1]^\top$ we get

$$\bar{\Delta}_R = \mathbf{1}^\top \Delta_R = 1bp \cdot \sum_k \frac{\partial V}{\partial R_k} \approx \frac{V(\bar{R} + 1bp \cdot \mathbf{1}) - V(\bar{R} - 1bp \cdot \mathbf{1})}{2} \quad \text{and}$$

$$\bar{\Gamma}_R = \mathbf{1}^\top \Gamma_R = [1bp]^2 \cdot \sum_k \frac{\partial^2 V}{\partial R_k^2} \approx V(\bar{R} + 1bp \cdot \mathbf{1}) - 2V(\bar{R}) + V(\bar{R} - 1bp \cdot \mathbf{1})$$

Vega is the sensitivity w.r.t. changes in market volatilities

Bucketed ATM Normal Volatility Vega

Denote $\bar{\sigma} = \left[\sigma_N^{k,l}\right]$ the matrix of market-implied At-the-money normal volatilites for expiries $k = 1, \ldots, q$ and swap terms $l = 1, \ldots, r$. Bucketed ATM Normal Volatility Vega of an instrument with model price $V = V(\bar{\sigma})$ is specified as

$$\text{Vega} = 1bp \cdot \left[\frac{\partial V}{\partial \sigma_N^{k,l}}\right]_{k=1,\ldots,q,\ l=1,\ldots,r}.$$

Parallel ATM Normal Volatility Vega

Parallel ATM Normal Volatility Vega represents sensitivity w.r.t. a parallel shift in the implied ATM swaption volatility surface. That is

$$\overline{\text{Vega}} = 1bp \cdot \mathbf{1}^\top \left[\text{Vega}\right] \mathbf{1} = 1bp \cdot \sum_{k,l} \frac{\partial V}{\partial \sigma_N^{k,l}} \approx \frac{V(\bar{\sigma} + 1bp \cdot \mathbf{1}\,\mathbf{1}^\top) - V(\bar{\sigma} - 1bp \cdot \mathbf{1}\,\mathbf{1}^\top)}{2}$$

- Volatility smile sensitivities are often specified in terms of Vanilla model parameter sensitivities

Outline

Crutial part of sensitivity calculation is evaluation or approximation of partial derivatives

Consider again general pricing function $V = V(p)$ in terms of a scalar parameter p. Assume differentiability of V w.r.t. p and sensitivity

$$\Delta V = \frac{dV(p)}{dp} \cdot \Delta p$$

Finite Difference Approximation

Finite difference approximation with step size h is

$$\frac{dV(p)}{dp} \approx \frac{V(p+h) - V(p)}{h} \text{ or } \frac{dV(p)}{dp} \approx \frac{V(p) - V(p-h)}{h} \quad \text{(one-sided), or}$$

$$\frac{dV(p)}{dp} \approx \frac{V(p+h) - V(p-h)}{2h} \quad \text{(two-sided)}$$

- Simple to implement and calculate (only pricing function evaluation)
- Typically for black-box pricing functions

We do a case study for European swaption Vega I

Recall pricing function

$$V^{\text{Swpt}} = Ann(t) \cdot \text{Bachelier}\left(S(t), K, \sigma\sqrt{T-t}, \phi\right)$$

with

$$\text{Bachelier}\left(F, K, \nu, \phi\right) = \nu \cdot \left[\Phi\left(h\right) \cdot h + \Phi'\left(h\right)\right], \quad h = \frac{\phi\left[F-K\right]}{\nu}.$$

First, analyse Bachelier formula. We get

$$\begin{aligned}\frac{d}{d\nu}\text{Bachelier}\left(\nu\right) &= \frac{\text{Bachelier}\left(\nu\right)}{\nu} + \nu\left[\left(\Phi'\left(h\right)h + \Phi\left(h\right)\right)\frac{dh}{d\nu} - \Phi'\left(h\right)h\frac{dh}{d\nu}\right] \\ &= \frac{\text{Bachelier}\left(\nu\right)}{\nu} + \nu\Phi\left(h\right)\frac{dh}{d\nu}.\end{aligned}$$

With $\frac{dh}{d\nu} = -\frac{h}{\nu}$ follows

$$\frac{d}{d\nu}\text{Bachelier}\left(\nu\right) = \Phi\left(h\right) \cdot h + \Phi'\left(h\right) - \Phi\left(h\right) \cdot h = \Phi'\left(h\right).$$

We do a case study for European swaption Vega II

Moreover, second derivative (Volga) becomes

$$\frac{d^2}{d\nu^2}\text{Bachelier}(\nu) = -h\Phi'(h)\frac{dh}{d\nu} = \frac{h^2}{\nu}\Phi'(h).$$

This gives for ATM options with $h = 0$ that

- Volga $\frac{d^2}{d\nu^2}\text{Bachelier}(\nu) = 0$
- ATM option price is approximately linear in volatility ν

Differentiating once again yields (we skip details)

$$\frac{d^3}{d\nu^3}\text{Bachelier}(\nu) = \left(h^2 - 3\right)\frac{h^2}{\nu^2}\Phi'(h)$$

It turns out that Volga has a maximum at moneyness

$$h = \pm\sqrt{3}.$$

We do a case study for European swaption Vega III

Swaption Vega becomes

$$\frac{d}{d\sigma} V^{\text{Swpt}} = Ann(t) \cdot \frac{d}{d\nu} \text{Bachelier}\,(\nu) \cdot \sqrt{T - t}$$

Test case

- Rates flat at 5%, implied normal volatilities flat at $100bp$
- 10y into 10y European payer swaption (call on swap rate)
- Strike at $5\% + 100bp \cdot \sqrt{10y} \cdot \sqrt{3} = 10.48\%$ (maximizing Volga)

What is the problem with finite difference approximation? I

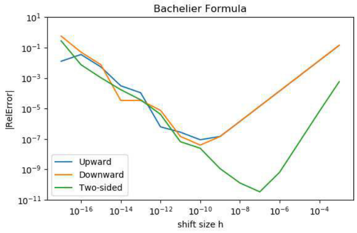
- There is a non-trivial trade-off between convergence and numerical accuracy
- We have analytical Vega formula from Bachelier formula and implied normal volatility

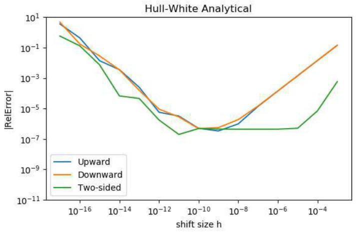
$$\text{Vega} = Ann(t) \cdot \Phi'(h) \cdot \sqrt{T-t}$$

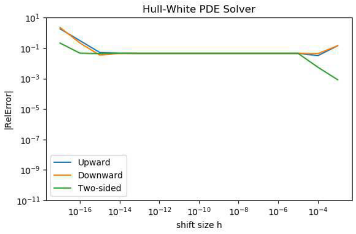
- Compare one-sided (upward and downward) and two-sided finite difference approximation Vega_{FD} using
  - Bachelier formula
  - Analytical Hull-White coupon bond option formula
  - Hull-White model via PDE solver (Crank-Nicolson, 101 grid points, 3 stdDevs wide, 1m time stepping)
  - Hull-White model via density integration (C^2-spline exact with break-even point, 101 grid points, 5 stdDevs wide)
- Compare absolute relative error (for all finite difference approximations)

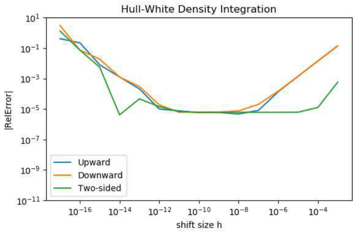
$$|\text{RelErr}| = \left[\frac{\text{Vega}_{FD}}{\text{Vega}} - 1\right]$$

What is the problem with finite difference approximation? II









Optimal choice of FD step size *h* is very problem-specific and depends on discretisation of numerical method

Outline

Derivative pricing usually involves model calibration

Consider swap pricing function V^{Swap} as a function of yield curve model parameters z, i.e.

$$V^{\text{Swap}} = V^{\text{Swap}}(z).$$

Model parameters z are itself derived from market quotes R for par swaps and FRAs. That is

$$z = z(R).$$

This gives mapping

$$R \mapsto z \mapsto V^{\text{Swap}} = V^{\text{Swap}}\left(z(R)\right).$$

Interest rate Delta becomes

$$\Delta_R = 1bp \cdot \underbrace{\frac{dV^{\text{Swap}}}{dz}\left(z(R)\right)}_{\text{Pricing}} \cdot \underbrace{\frac{dz}{dR}\left(R\right)}_{\text{Calibration}}$$

- Suppose a large portfolio of swaps
  - Calibration Jacobian $\frac{dz(R)}{dR}$ is the same for all swaps in portfolio
  - Save computational effort by pre-calculating and storing Jacobian
- Brute-force finite difference approximation of Jacobian may become inaccurate due to numerical scheme for calibration/optimisation

Can we calculate calibration Jacobian more efficiently?

Theorem (Implicit Function Theorem)

Let $\mathcal{H} : \mathbb{R}^q \times \mathbb{R}^r \to \mathbb{R}^q$ be a continuously differentiable function with $\mathcal{H}(\bar{z}, \bar{R}) = 0$ for some pair $(\bar{z}, \bar{R})$. If the Jacobian

$$J_z = \frac{d\mathcal{H}}{dz}(\bar{z}, \bar{R})$$

is invertible, then there exists an open domain $\mathcal{U} \subset \mathbb{R}^r$ with $\bar{R} \in \mathcal{U}$ and a continuously differentiable function $g : \mathcal{U} \to \mathbb{R}^q$ with

$$\mathcal{H}\left(g(R), R\right) = 0 \quad \forall R \in \mathcal{U}.$$

Moreover, we get for the Jacobian of g that

$$\frac{dg(R)}{dR} = -\left[\frac{d\mathcal{H}}{dz}(g(R), R)\right]^{-1} \left[\frac{d\mathcal{H}}{dR}(g(R), R)\right].$$

Proof.

See Analysis.

$\square$

How does Implicit Function Theorem help for sensitivity calculation? I

- Consider $\mathcal{H}(z, R)$ the q-dimensional objective function of yield curve calibration problem
  - $z = [z_1, \ldots, z_q]^\top$ yield curve parameters (e.g. zero rates or forward rates)
  - $R = [R_1, \ldots, R_q]^\top$ market quotes (par rates) for swaps and FRAs
  - set $r = q$, i.e. same number of market quotes as model parameters
- Re-formulate calibration helpers slightly such that

$$\mathcal{H}_k(z, R) = \text{ModelRate}_k(z) - R_k$$

  - e.g. for swaps model-implied par swap rate becomes

$$\text{ModelRate}_k(z) = \frac{\sum_{j=1}^{m_k} L^\delta(0, \tilde{T}_{j-1}, \tilde{T}_{j-1} + \delta) \cdot \tilde{\tau}_j \cdot P(t, \tilde{T}_j)}{\sum_{i=1}^{n_k} \tau_i \cdot P(0, T_i)}$$

How does Implicit Function Theorem help for sensitivity calculation? II

If pair $(\bar{z}, \bar{R})$ solves calibration problem $\mathcal{H}(\bar{z}, \bar{R}) = 0$ and $\frac{d\mathcal{H}}{dz}(\bar{z}, \bar{R})$ is invertible, then there exists a function

$$z = z(R)$$

in a vicinity of $\bar{R}$ and

$$\frac{dz}{dR}(R) = -\left[\frac{d\mathcal{H}}{dz}(g(R), R)\right]^{-1}\left[\frac{d\mathcal{H}}{dR}(g(R), R)\right].$$

Re-formulation of calibration helpers gives

$$\frac{d\mathcal{H}}{dz}(g(R), R) = \begin{bmatrix} \frac{d}{dz}\text{ModelRate}_1(z) \\ \vdots \\ \frac{d}{dz}\text{ModelRate}_q(z) \end{bmatrix}, \quad \text{and}$$

$$\frac{d\mathcal{H}}{dR}(g(R), R) = \begin{bmatrix} -1 & & \\ & \ddots & \\ & & -1 \end{bmatrix}$$

How does Implicit Function Theorem help for sensitivity calculation? III

Consequently

$$\frac{dz}{dR}(R) = \left[\frac{d\mathcal{H}}{dz}(g(R),R)\right]^{-1} = \begin{bmatrix} \frac{d}{dz}\text{ModelRate}_1(z) \\ \vdots \\ \frac{d}{dz}\text{ModelRate}_q(z) \end{bmatrix}^{-1}.$$

We get Jacobian method for risk calculation

$$\Delta_R = 1bp \cdot \underbrace{\frac{dV^{\text{Swap}}}{dz}(z(R))}_{\text{Pricing}} \cdot \underbrace{\begin{bmatrix} \frac{d}{dz}\text{ModelRate}_1(z) \\ \vdots \\ \frac{d}{dz}\text{ModelRate}_q(z) \end{bmatrix}^{-1}}_{\text{Calibration}}$$

- Requires only sensitivities w.r.t. model parameters
- Reference market intruments/rates R_k can also be chosen independent of original calibration problem
- Calibration Jacobian and matrix inversion can be pre-computed and stored

We can adapt Jacobian method to Vega calculation as well I

Bermudan swaption is determined via mapping

$$\underbrace{\left[\sigma_N^1, \ldots \sigma_N^{\bar{k}}\right]}_{\text{market-impl. normal vols}} \mapsto \underbrace{\left[\sigma^1, \ldots \sigma^{\bar{k}}\right]}_{\text{HW short rate vols}} \mapsto V^{\text{Berm}}.$$

Assign volatility calibration helpers

$$\mathcal{H}_k\left(\sigma, \sigma_N\right) = \underbrace{V_k^{\text{CBO}}(\sigma)}_{\text{Model}[\sigma]} - \underbrace{V_k^{\text{Swpt}}(\sigma_N^k)}_{\text{Market}\left(\sigma_N^k\right)}$$

- $V_k^{\text{CBO}}(\sigma)$ Hull-White model price of kth co-terminal European swaption represented as coupon bond option
- $V_k^{\text{Swpt}}(\sigma_N^k)$ Bachelier formula to calculate market price for kth co-terminal European swaption from given normal volatility σ_N^k

We can adapt Jacobian method to Vega calculation as well II

Implicit Function Theorem yields

$$
\begin{aligned}
\frac{d\sigma}{d\sigma_N} &= -\left[\frac{d\mathcal{H}}{d\sigma}\left(\sigma\left(\sigma_N\right),\sigma_N\right)\right]^{-1}\left[\frac{d\mathcal{H}}{d\sigma_N}\left(\sigma\left(\sigma_N\right),\sigma_N\right)\right] \\
&= \left[\frac{d}{d\sigma}\text{Model}[\sigma]\right]^{-1}\begin{bmatrix} \frac{d}{d\sigma_N}V_1^{\text{Swpt}}(\sigma_N^1) & & \\ & \ddots & \\ & & \frac{d}{d\sigma_N}V_{\bar{k}}^{\text{Swpt}}(\sigma_N^{\bar{k}}) \end{bmatrix}
\end{aligned}
$$

- $\frac{d}{d\sigma}\text{Model}[\sigma]$ are Hull-White model Vega(s) of co-terminal European swaptions
- $\frac{d}{d\sigma_N}V_k^{\text{Swpt}}(\sigma_N^k)$ are Bachelier or market Vega(s) of co-terminal European swaptions

Bermudan Vega becomes

$$
\frac{d}{d\sigma_N}V^{\text{Berm}} = \frac{d}{d\sigma}V^{\text{Berm}} \cdot \left[\frac{d}{d\sigma}\text{Model}[\sigma]\right]^{-1} \cdot \frac{d}{d\sigma_N}\text{Market}\left(\sigma_N^k\right).
$$

Outline

What is the idea behind Algorithmic Differentiation (AD)

- AD covers principles and techniques to augment computer models or programs
- Calculate sensitivities of output variables with respect to inputs of a model
- Compute numerical values rather than symbolic expressions
- Sensitivities are exact up to machine precision (no rounding/cancellation errors as in FD)
- Apply chain rule of differentiation to operations like `+`, `*`, and intrinsic functions like `exp(.)`

Functions are represented as Evaluation Procedures consisting of a sequence of elementary operations

Example: Black Formula

$$\text{Black}(\cdot) = \omega\left[F\Phi(\omega d_1) - K\Phi(\omega d_2)\right]$$

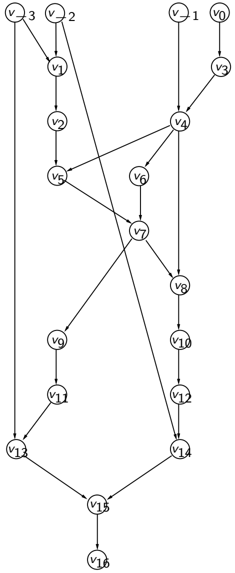
with $d_{1,2} = \frac{\log(F/K)}{\sigma\sqrt{\tau}} \pm \frac{\sigma\sqrt{\tau}}{2}$

- Inputs F, K, σ, τ
- Discrete parameter $\omega \in \{-1, 1\}$
- Output Black$(\cdot)$

$$\begin{array}{lcllcl}
v_{-3} & = & x_1 = F & & \\
v_{-2} & = & x_2 = K & & \\
v_{-1} & = & x_3 = \sigma & & \\
v_0 & = & x_4 = \tau & & \\
\hline
v_1 & = & v_{-3}/v_{-2} & \equiv & f_1(v_{-3}, v_{-2}) \\
v_2 & = & \log(v_1) & \equiv & f_2(v_1) \\
v_3 & = & \sqrt{v_0} & \equiv & f_3(v_0) \\
v_4 & = & v_{-1} \cdot v_3 & \equiv & f_4(v_{-1}, v_3) \\
v_5 & = & v_2/v_4 & \equiv & f_5(v_2, v_4) \\
v_6 & = & 0.5 \cdot v_4 & \equiv & f_6(v_4) \\
v_7 & = & v_5 + v_6 & \equiv & f_7(v_5, v_6) \\
v_8 & = & v_7 - v_4 & \equiv & f_8(v_7, v_4) \\
v_9 & = & \omega \cdot v_7 & \equiv & f_9(v_7) \\
v_{10} & = & \omega \cdot v_8 & \equiv & f_{10}(v_8) \\
v_{11} & = & \Phi(v_9) & \equiv & f_{11}(v_9) \\
v_{12} & = & \Phi(v_{10}) & \equiv & f_{12}(v_{10}) \\
v_{13} & = & v_{-3} \cdot v_{11} & \equiv & f_{13}(v_{-3}, v_{11}) \\
v_{14} & = & v_{-2} \cdot v_{12} & \equiv & f_{14}(v_{-2}, v_{12}) \\
v_{15} & = & v_{13} - v_{14} & \equiv & f_{15}(v_{13}, v_{14}) \\
v_{16} & = & \omega \cdot v_{15} & \equiv & f_{16}(v_{15}) \\
\hline
y_1 & = & v_{16} & & \\
\end{array}$$

Alternative representation is Directed Acyclic Graph (DAG)

$$
\begin{array}{lclcl}
v_{-3} & = & x_1 = F & & \\
v_{-2} & = & x_2 = K & & \\
v_{-1} & = & x_3 = \sigma & & \\
v_0 & = & x_4 = \tau & & \\
\hline
v_1 & = & v_{-3}/v_{-2} & \equiv & f_1(v_{-3}, v_{-2}) \\
v_2 & = & \log(v_1) & \equiv & f_2(v_1) \\
v_3 & = & \sqrt{v_0} & \equiv & f_3(v_0) \\
v_4 & = & v_{-1} \cdot v_3 & \equiv & f_4(v_{-1}, v_3) \\
v_5 & = & v_2/v_4 & \equiv & f_5(v_2, v_4) \\
v_6 & = & 0.5 \cdot v_4 & \equiv & f_6(v_4) \\
v_7 & = & v_5 + v_6 & \equiv & f_7(v_5, v_6) \\
v_8 & = & v_7 - v_4 & \equiv & f_8(v_7, v_4) \\
v_9 & = & \omega \cdot v_7 & \equiv & f_9(v_7) \\
v_{10} & = & \omega \cdot v_8 & \equiv & f_{10}(v_8) \\
v_{11} & = & \Phi(v_9) & \equiv & f_{11}(v_9) \\
v_{12} & = & \Phi(v_{10}) & \equiv & f_{12}(v_{10}) \\
v_{13} & = & v_{-3} \cdot v_{11} & \equiv & f_{13}(v_{-3}, v_{11}) \\
v_{14} & = & v_{-2} \cdot v_{12} & \equiv & f_{14}(v_{-2}, v_{12}) \\
v_{15} & = & v_{13} - v_{14} & \equiv & f_{15}(v_{13}, v_{14}) \\
v_{16} & = & \omega \cdot v_{15} & \equiv & f_{16}(v_{15}) \\
\hline
y_1 & = & v_{16} & &
\end{array}
$$



Evaluation Procedure can be formalized to make it more tractable

Definition (Evaluation Procedure)

Suppose $F : \mathbb{R}^n \to \mathbb{R}^m$ and $f_i : \mathbb{R}^{n_i} \to \mathbb{R}^{m_i}$. The relation $j \prec i$ denotes that $v_i \in \mathbb{R}$ depends directly on $v_j \in \mathbb{R}$. If for all $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ with $y = F(x)$ holds that

$$\begin{array}{rcll} v_{i-n} & = & x_i & i = 1, \ldots, n \\ v_i & = & f_i(v_j)_{j \prec i} & i = 1, \ldots, l \\ y_{m-i} & = & v_{l-i} & i = m-1, \ldots, 0, \end{array}$$

then we call this sequence of operations an evaluation procedure of F with elementary operations f_i. We assume differentiability of all elementary operations f_i $(i = 1, \ldots, l)$. Then the resulting function F is also differentiable.

- Abbreviate $u_i = (v_j)_{j \prec i} \in \mathbb{R}^{n_i}$ the collection of arguments of the operation f_i
- Then we may also write
$$v_i = f_i(u_i)$$

Forward mode of AD calculates tangents

- In addition to function evaluation $v_i = f_i(u_i)$ evaluate derivative

$$\dot{v}_i \; = \; \sum_{j \prec i} \frac{\partial}{\partial v_j} f_i(u_i) \cdot \dot{v}_j$$

Forward Mode or Tangent Mode of AD

Use abbreviations $\dot{u}_i = (\dot{v}_j)_{j \prec i}$ and $\dot{f}_i(u_i, \dot{u}_i) = f_i'(u_i) \cdot \dot{u}_i$. The Forward Mode of AD is the augmented evaluation procedure

$$\begin{array}{rcll} [v_{i-n}, \dot{v}_{i-n}] & = & [x_i, \dot{x}_i] & i = 1, \ldots, n \\ [v_i, \dot{v}_i] & = & \left[f_i(u_i), \dot{f}_i(u_i, \dot{u}_i)\right] & i = 1, \ldots, l \\ [y_{m-i}, \dot{y}_{m-i}] & = & [v_{l-i}, \dot{v}_{l-i}] & i = m-1, \ldots, 0. \end{array}$$

Here, the initializing derivative values $\dot{x}_{i-n}$ for $i = 1 \ldots n$ are given and determine the direction of the tangent.

- With $\dot{x} = (\dot{x}_i) \in \mathbb{R}^n$ and $\dot{y} = (\dot{y}_i) \in \mathbb{R}^m$, the forward mode of AD evaluates

$$\dot{y} \; = \; F'(x)\dot{x}$$

- Computational effort is approx. 2.5 function evaluations of F

Black formula Forward Mode evaluation procedure...

$$
\begin{array}{lcl|lcl}
v_{-3} & = & x_1 = F & \dot{v}_{-3} & = & 0 \\
v_{-2} & = & x_2 = K & \dot{v}_{-2} & = & 0 \\
v_{-1} & = & x_3 = \sigma & \dot{v}_{-1} & = & 1 \\
v_0 & = & x_4 = \tau & \dot{v}_0 & = & 0 \\
\hline
v_1 & = & v_{-3}/v_{-2} & \dot{v}_1 & = & \dot{v}_{-3}/v_{-2} - v_1 \cdot \dot{v}_{-2}/v_{-2} \\
v_2 & = & \log(v_1) & \dot{v}_2 & = & \dot{v}_1/v_1 \\
v_3 & = & \sqrt{v_0} & \dot{v}_3 & = & 0.5 \cdot \dot{v}_0/v_3 \\
v_4 & = & v_{-1} \cdot v_3 & \dot{v}_4 & = & \dot{v}_{-1} \cdot v_3 + v_{-1} \cdot \dot{v}_3 \\
v_5 & = & v_2/v_4 & \dot{v}_5 & = & \dot{v}_2/v_4 - v_5 \cdot \dot{v}_4/v_4 \\
v_6 & = & 0.5 \cdot v_4 & \dot{v}_6 & = & 0.5 \cdot \dot{v}_4 \\
v_7 & = & v_5 + v_6 & \dot{v}_7 & = & \dot{v}_5 + \dot{v}_6 \\
v_8 & = & v_7 - v_4 & \dot{v}_8 & = & \dot{v}_7 - \dot{v}_4 \\
v_9 & = & \omega \cdot v_7 & \dot{v}_9 & = & \omega \cdot \dot{v}_7 \\
v_{10} & = & \omega \cdot v_8 & \dot{v}_{10} & = & \omega \cdot \dot{v}_8 \\
v_{11} & = & \Phi(v_9) & \dot{v}_{11} & = & \phi(v_9) \cdot \dot{v}_9 \\
v_{12} & = & \Phi(v_{10}) & \dot{v}_{12} & = & \phi(v_{10}) \cdot \dot{v}_{10} \\
v_{13} & = & v_{-3} \cdot v_{11} & \dot{v}_{13} & = & \dot{v}_{-3} \cdot v_{11} + v_{-3} \cdot \dot{v}_{11} \\
v_{14} & = & v_{-2} \cdot v_{12} & \dot{v}_{14} & = & \dot{v}_{-2} \cdot v_{12} + v_{-2} \cdot \dot{v}_{12} \\
v_{15} & = & v_{13} - v_{14} & \dot{v}_{15} & = & \dot{v}_{13} - \dot{v}_{14} \\
v_{16} & = & \omega \cdot v_{15} & \dot{v}_{16} & = & \omega \cdot \dot{v}_{15} \\
\hline
y_1 & = & v_{16} & \dot{y}_1 & = & \dot{v}_{16}
\end{array}
$$

Reverse Mode of AD calculates adjoints I

- Forward Mode calculates derivatives and applies chain rule in the same order as function evaluation
- Reverse Mode of AD applies chain rule in reverse order of function evaluation
- Define auxiliary derivative values $\bar{v}_j$ and assume initialisation $\bar{v}_j = 0$ before reverse mode evaluation
- For each elementary operation f_i and all intermediate variables v_j with $j \prec i$, evaluate

$$\bar{v}_j \mathrel{+}= \bar{v}_i \cdot \frac{\partial}{\partial v_j} f_i(u_i)$$

- In other words, for each arguments of f_i the partial derivative is derived

Reverse Mode of AD calculates adjoints II

Reverse Mode or Adjoint Mode of AD

Denoting $\bar{u}_i = (\bar{v}_j)_{j \prec i} \in \mathbb{R}^{n_i}$ and $\bar{f}_i(u_i, \bar{v}_i) = \bar{v}_i \cdot f_i'(u_i)$, the *incremental reverse mode of AD* is given by the evaluation procedure

$$
\begin{array}{rcll}
v_{i-n} & = & x_i & i = 1, \dots, n \\
v_i & = & f_i(v_j)_{j \prec i} & i = 1, \dots, l \\
y_{m-i} & = & v_{l-i} & i = m-1, \dots, 0 \\
\hline
\bar{v}_i & = & \bar{y}_i & i = 0, \dots, m-1 \\
\bar{u}_i & += & \bar{f}_i(u_i, \bar{v}_i) & i = l, \dots, 1 \\
\bar{x}_i & = & \bar{v}_i & i = n, \dots, 1.
\end{array}
$$

Here, all intermediate variables v_i are assigned only once. The initializing values $\bar{y}_i$ are given and represent a weighting of the dependent variables y_i.

- Vector $\bar{y} = (\bar{y}_i)$ can also be interpreted as normal vector of a hyperplane in the range of F
- With $\bar{y} = (\bar{y}_i)$ and $\bar{x} = (\bar{x}_i)$, reverse mode of AD yields

$$\bar{x}^T = \nabla \left[\bar{y}^T F(x) \right] = \bar{y}^T F'(x)$$

- Computational effort is approx. 4 function evaluations of F

Black formula Reverse Mode evaluation procedure... I

$$
\begin{aligned}
v_{-3} &= x_1 = F \\
v_{-2} &= x_2 = K \\
v_{-1} &= x_3 = \sigma \\
v_0 &= x_4 = \tau \\
\hline
v_1 &= v_{-3}/v_{-2} \\
v_2 &= \log(v_1) \\
v_3 &= \sqrt{v_0} \\
v_4 &= v_{-1} \cdot v_3 \\
v_5 &= v_2/v_4 \\
v_6 &= 0.5 \cdot v_4 \\
v_7 &= v_5 + v_6 \\
v_8 &= v_7 - v_4 \\
v_9 &= \omega \cdot v_7 \\
v_{10} &= \omega \cdot v_8 \\
v_{11} &= \Phi(v_9) \\
v_{12} &= \Phi(v_{10}) \\
v_{13} &= v_{-3} \cdot v_{11} \\
v_{14} &= v_{-2} \cdot v_{12} \\
v_{15} &= v_{13} - v_{14} \\
v_{16} &= \omega \cdot v_{15} \\
\hline
y_1 &= v_{16} \\
\bar{v}_{16} &= \bar{y}_1 = 1 \\
\hline
&\vdots
\end{aligned}
$$

Black formula Reverse Mode evaluation procedure... II

$$\vdots$$

$$y_1 = v_{16}$$
$$\bar{v}_{16} = \bar{y}_1 = 1$$

$$\bar{v}_{15} \mathrel{+}= \omega \cdot \bar{v}_{16}$$
$$\bar{v}_{13} \mathrel{+}= \bar{v}_{15}; \quad \bar{v}_{14} \mathrel{+}= (-1) \cdot \bar{v}_{15}$$
$$\bar{v}_{-2} \mathrel{+}= v_{12} \cdot \bar{v}_{14}; \quad \bar{v}_{12} \mathrel{+}= v_{-2} \cdot \bar{v}_{14}$$
$$\bar{v}_{-3} \mathrel{+}= v_{11} \cdot \bar{v}_{13}: \quad \bar{v}_{11} \mathrel{+}= v_{-3} \cdot \bar{v}_{13}$$
$$\bar{v}_{10} \mathrel{+}= \phi(v_{10}) \cdot \bar{v}_{12}$$
$$\bar{v}_{9} \mathrel{+}= \phi(v_{9}) \cdot \bar{v}_{11}$$
$$\bar{v}_{8} \mathrel{+}= \omega \cdot \bar{v}_{10}$$
$$\bar{v}_{7} \mathrel{+}= \omega \cdot \bar{v}_{9}$$
$$\bar{v}_{7} \mathrel{+}= \bar{v}_{8}; \quad \bar{v}_{4} \mathrel{+}= (-1) \cdot \bar{v}_{8}$$
$$\bar{v}_{5} \mathrel{+}= \bar{v}_{7}; \quad \bar{v}_{6} \mathrel{+}= \bar{v}_{7}$$
$$\bar{v}_{4} \mathrel{+}= 0.5 \cdot \bar{v}_{6}$$
$$\bar{v}_{2} \mathrel{+}= \bar{v}_{5}/v_{4}; \quad \bar{v}_{4} \mathrel{+}= (-1) \cdot v_{5} \cdot \bar{v}_{5}/v_{4}$$
$$\bar{v}_{-1} \mathrel{+}= v_{3} \cdot \bar{v}_{4}; \quad \bar{v}_{3} \mathrel{+}= v_{-1} \cdot \bar{v}_{4}$$
$$\bar{v}_{0} \mathrel{+}= 0.5 \cdot \bar{v}_{3}/v_{3}$$
$$\bar{v}_{1} \mathrel{+}= \bar{v}_{2}/v_{1}$$
$$\bar{v}_{-3} \mathrel{+}= \bar{v}_{1}/v_{-2}; \quad \bar{v}_{-2} \mathrel{+}= (-1) \cdot v_{1} \cdot \bar{v}_{1}/v_{-2}$$

$$\bar{\tau} = \bar{x}_4 = \bar{v}_0$$
$$\bar{\sigma} = \bar{x}_3 = \bar{v}_{-1}$$
$$\bar{K} = \bar{x}_2 = \bar{v}_{-2}$$
$$\bar{F} = \bar{x}_1 = \bar{v}_{-3}$$

We summarize the properties of Forward and Reverse Mode

Forward Mode

$$\dot{y} = F'(x)\dot{x}$$

- Approx. 2.5 function evaluations
- Computational effort independent of number of output variables (dimension of y)
- Chain rule in same order as computation
- Memory consumption in order of function evaluation

Reverse Mode

$$\bar{x}^T = \bar{y}^T F'(x)$$

- Approx. 4 function evaluations
- Computational effort independent of number of input variables (dimension of x)
- Chain rule in reverse order of computation
- Requires storage of all intermediate results (or re-computation)
- Memory consumption/management key challange for implementations

- Computational effort can be improved by AD vector mode
- Reverse Mode memory consumption can be managed via checkpointing techniques

How is AD applied in practice?

- Typically, you don't want to differentiate all your source code by hand
- Tools help augmenting existing programs for tangent and adjoint computations

Source Code Transformation

- Applied to the model code in compiler fashion
- Generate AD model as new source code
- Original code may need to be adapted slightly to meet capabilities of AD tool

Some example C++ tools

ADIC2, dcc, TAPENADE

Operator Overloading

- provide new (active) data type
- Overload all relevant operators/ functions with sensitivity aware arithmetic
- AD model derived by changing intrinsic to active data type

ADOL-C, dco/c++, ADMB/AUTODIF

- There are also tools for Python and other lamguages

More details at `autodiff.org`

There is quite some literature on AD and its application in finance

Standard textbook on AD

- A. Griewank and A. Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation - 2nd ed.*
  SIAM, 2008

Recent practitioner's textbook

- U. Naumann. *The Art of Differentiating Computer Programs: An Introduction to Algorithmic Differentiation.*
  SIAM, 2012

One of the first and influencial papers for AD application in finance

- M. Giles and P. Glasserman. Smoking adjoints: fast monte carlo greeks.
  Risk, January 2006

Contact

Dr. Sebastian Schlenkrich
Office: RUD25, R 1.211
Mail: sebastian.schlenkrich@hu-berlin.de

d-fine GmbH
Mobile: +49-162-263-1525
Mail: sebastian.schlenkrich@d-fine.de