Part VII

Sensitivity Calculation
Outline

Introduction to Sensitivity Calculation

Finite Difference Approximation for Sensitivities

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation
Outline

Introduction to Sensitivity Calculation

Finite Difference Approximation for Sensitivities

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation
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 $\Delta t$ and returns

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

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

$$\text{VaR}_\alpha = \inf \{ q \; s.t. \; \mathbb{P}\{\Delta \pi(t) \leq q \mid \pi(t)\} > \alpha \}.$$

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

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

- 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 $\pi$ 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

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 $\Delta V$ becomes

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

Typical scaling (or risk factor shift sizes) $\Delta p$ are

- $1\text{bp}$ for interest rate shifts,
- $1\text{bp}$ 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.

**Bucketed Delta and Gamma**

Let $\bar{R} = [R_k]_{k=1, \ldots, q}$ 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

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

Bucketed Gamma is calculated as

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

**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. simultaneous shifts of all market rates of a yield curve. With $\mathbf{1} = [1, \ldots, 1]^\top$ we get

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

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 $\bar{\sigma} = [\sigma_{N}^{k,l}]$ the matrix of market-implied At-the-money normal volatilities 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

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

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

$$\text{Parallel Vega} = 1 \text{bp} \cdot 1^\top [\text{Vega}] 1 = 1 \text{bp} \cdot \sum_{k,l} \frac{\partial V}{\partial \sigma_{N}^{k,l}} \approx \frac{V(\bar{\sigma} + 1 \text{bp} \cdot 1 1^\top) - V(\bar{\sigma} - 1 \text{bp} \cdot 1 1^\top)}{2}.$$

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

Introduction to Sensitivity Calculation

Finite Difference Approximation for Sensitivities

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation
Crucial 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

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

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

- 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

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

with

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

First, analyse Bachelier formula. We get

\[
\frac{d}{d\nu} \text{Bachelier}\left(\nu\right) = \frac{\text{Bachelier}\left(\nu\right)}{\nu} + \nu \left[ \left(\Phi'(h) \cdot h + \Phi(h)\right) \frac{dh}{d\nu} - \Phi'(h) \cdot h \frac{dh}{d\nu} \right]
\]

\[ = \frac{\text{Bachelier}\left(\nu\right)}{\nu} + \nu \Phi(h) \frac{dh}{d\nu}. \]

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

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

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

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

Test case

- Rates flat at 5\%, implied normal volatilities flat at 100bp.
- 10y into 10y European payer swaption (call on swap rate).
- Strike at \(5\% + 100\text{bp} \cdot \sqrt{10\text{y}} \cdot \sqrt{3} = 10.48\%\) (maximizing Volga).
What is the problem with finite difference approximation?

- There is a non-trivial trade-off between convergence and numerical accuracy.
- We have analytical Vega formula from Bachelier formula and implied normal volatility:
  \[ \text{Vega} = \text{Ann}(t) \cdot \Phi'(h) \cdot \sqrt{T - t}. \]
- Compare one-sided (upward and downward) and two-sided finite difference approximation \(\text{Vega}_{FD}\) 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?

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

Introduction to Sensitivity Calculation

Finite Difference Approximation for Sensitivities

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation
Derivative pricing usually involves model calibration

Consider swap pricing function $V^{\text{Swap}}$ 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

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

Interest rate Delta becomes

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

- 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 $U \subset \mathbb{R}^r$ with $\bar{R} \in U$ and a continuously differentiable function $g : U \to \mathbb{R}^q$ with

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

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.
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.

Reformulate 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

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

Reformulation of calibration helpers gives

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

$$\frac{d\mathcal{H}}{dR}(g(R), R) = \begin{bmatrix} -1 \\ \vdots \\ -1 \end{bmatrix}. $$
How does Implicit Function Theorem help for sensitivity calculation? III

Consequently

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

We get Jacobian method for risk calculation

\[
\Delta_R = 1 \text{bp} \cdot \frac{dV^{\text{Swap}}}{dz} (z(R)) \cdot \begin{bmatrix}
\frac{d}{dz} \text{ModelRate}_1(z) \\
\vdots \\
\frac{d}{dz} \text{ModelRate}_q(z)
\end{bmatrix}^{-1}.
\]

- Requires only sensitivities w.r.t. model parameters.
- Reference market instruments/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

Bermudan swaption is determined via mapping

\[
\begin{bmatrix}
\sigma_1^N, \ldots, \sigma_k^N
\end{bmatrix}
\quad \mapsto \quad
\begin{bmatrix}
\sigma_1, \ldots, \sigma_k
\end{bmatrix}
\quad \mapsto \quad V^{\text{Berm}}.
\]

Assign volatility calibration helpers

\[ H_k(\sigma, \sigma_N) = \underbrace{V^\text{CBO}_k(\sigma)}_{\text{Model}[\sigma]} - \underbrace{V^\text{Swpt}_k(\sigma_N^k)}_{\text{Market}(\sigma_N^k)}. \]

- \( V^\text{CBO}_k(\sigma) \) Hull-White model price of \( k \)th co-terminal European swaption represented as coupon bond option.
- \( V^\text{Swpt}_k(\sigma_N^k) \) Bachelier formula to calculate market price for \( k \)th co-terminal European swaption from given normal volatility \( \sigma_N^k \).
We can adapt Jacobian method to Vega calculation as well

Implicit Function Theorem yields

\[
\frac{d\sigma}{d\sigma_N} = - \left[ \frac{dH}{d\sigma} \left( \sigma(\sigma_N), \sigma_N \right) \right]^{-1} \left[ \frac{dH}{d\sigma_N} \left( \sigma(\sigma_N), \sigma_N \right) \right]
\]

\[
= \left[ \frac{d}{d\sigma} \text{Model}[\sigma] \right]^{-1} \begin{bmatrix}
\frac{d}{d\sigma_N} V_{1}^{Swpt} (\sigma_{1}^N) \\
\vdots \\
\frac{d}{d\sigma_N} V_{k}^{Swpt} (\sigma_{k}^N)
\end{bmatrix}
\]

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

Bermudan Vega becomes

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

Introduction to Sensitivity Calculation

Finite Difference Approximation for Sensitivities

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation
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** \( F, K, \sigma, \tau \)
- **Discrete parameter** \( \omega \in \{-1, 1\} \)
- **Output** \( \text{Black}(\cdot) \)

\[
\begin{align*}
\nu_{-3} &= x_1 = F \\
\nu_{-2} &= x_2 = K \\
\nu_{-1} &= x_3 = \sigma \\
\nu_0 &= x_4 = \tau \\
\nu_1 &= \frac{\nu_{-3}}{\nu_{-2}} \quad \equiv \quad f_1(\nu_{-3}, \nu_{-2}) \\
\nu_2 &= \log(\nu_1) \quad \equiv \quad f_2(\nu_1) \\
\nu_3 &= \sqrt{\nu_0} \quad \equiv \quad f_3(\nu_0) \\
\nu_4 &= \nu_{-1} \cdot \nu_3 \quad \equiv \quad f_4(\nu_{-1}, \nu_3) \\
\nu_5 &= \nu_2 / \nu_4 \quad \equiv \quad f_5(\nu_2, \nu_4) \\
\nu_6 &= 0.5 \cdot \nu_4 \quad \equiv \quad f_6(\nu_4) \\
\nu_7 &= \nu_5 + \nu_6 \quad \equiv \quad f_7(\nu_5, \nu_6) \\
\nu_8 &= \nu_7 - \nu_4 \quad \equiv \quad f_8(\nu_7, \nu_4) \\
\nu_9 &= \omega \cdot \nu_7 \quad \equiv \quad f_9(\nu_7) \\
\nu_{10} &= \omega \cdot \nu_8 \quad \equiv \quad f_{10}(\nu_8) \\
\nu_{11} &= \Phi(\nu_9) \quad \equiv \quad f_{11}(\nu_9) \\
\nu_{12} &= \Phi(\nu_{10}) \quad \equiv \quad f_{12}(\nu_{10}) \\
\nu_{13} &= \nu_{-3} \cdot \nu_{11} \quad \equiv \quad f_{13}(\nu_{-3}, \nu_{11}) \\
\nu_{14} &= \nu_{-2} \cdot \nu_{12} \quad \equiv \quad f_{14}(\nu_{-2}, \nu_{12}) \\
\nu_{15} &= \nu_{13} - \nu_{14} \quad \equiv \quad f_{15}(\nu_{13}, \nu_{14}) \\
\nu_{16} &= \omega \cdot \nu_{15} \quad \equiv \quad f_{16}(\nu_{15}) \\
\nu_1 &= \nu_{16}
\end{align*}
\]
Alternative representation is Directed Acyclic Graph (DAG)

\[
\begin{align*}
\nu_{-3} &= x_1 = F \\
\nu_{-2} &= x_2 = K \\
\nu_{-1} &= x_3 = \sigma \\
\nu_0 &= x_4 = \tau \\
\nu_1 &= \nu_{-3}/\nu_{-2} \equiv f_1(\nu_{-3}, \nu_{-2}) \\
\nu_2 &= \log(\nu_1) \equiv f_2(\nu_1) \\
\nu_3 &= \sqrt{\nu_0} \equiv f_3(\nu_0) \\
\nu_4 &= \nu_{-1} \cdot \nu_3 \equiv f_4(\nu_{-1}, \nu_3) \\
\nu_5 &= \nu_2/\nu_4 \equiv f_5(\nu_2, \nu_4) \\
\nu_6 &= 0.5 \cdot \nu_4 \equiv f_6(\nu_4) \\
\nu_7 &= \nu_5 + \nu_6 \equiv f_7(\nu_5, \nu_6) \\
\nu_8 &= \nu_7 - \nu_4 \equiv f_8(\nu_7, \nu_4) \\
\nu_9 &= \omega \cdot \nu_7 \equiv f_9(\nu_7) \\
\nu_{10} &= \omega \cdot \nu_8 \equiv f_{10}(\nu_8) \\
\nu_{11} &= \Phi(\nu_9) \equiv f_{11}(\nu_9) \\
\nu_{12} &= \Phi(\nu_{10}) \equiv f_{12}(\nu_{10}) \\
\nu_{13} &= \nu_{-3} \cdot \nu_{11} \equiv f_{13}(\nu_{-3}, \nu_{11}) \\
\nu_{14} &= \nu_{-2} \cdot \nu_{12} \equiv f_{14}(\nu_{-2}, \nu_{12}) \\
\nu_{15} &= \nu_{13} - \nu_{14} \equiv f_{15}(\nu_{13}, \nu_{14}) \\
\nu_{16} &= \omega \cdot \nu_{15} \equiv f_{16}(\nu_{15}) \\
y_1 &= \nu_{16}
\end{align*}
\]
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

$$v_{i-n} = x_i, \quad i = 1, \ldots, n$$
$$v_i = f_i(v_{j \prec i}) \quad i = 1, \ldots, l$$
$$y_{m-i} = v_{l-i}, \quad i = m - 1, \ldots, 0,$$

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

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

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 $F$. 

\[p. 494\]
Black formula Forward Mode evaluation procedure...

\[ \begin{align*}
\nu_{-3} &= x_1 = F & \dot{\nu}_{-3} &= 0 \\
\nu_{-2} &= x_2 = K & \dot{\nu}_{-2} &= 0 \\
\nu_{-1} &= x_3 = \sigma & \dot{\nu}_{-1} &= 1 \\
\nu_0 &= x_4 = \tau & \dot{\nu}_0 &= 0 \\
\nu_1 &= \frac{\nu_{-3}}{\nu_{-2}} & \dot{\nu}_1 &= \frac{\nu_{-3}}{\nu_{-2}} - \nu_1 \cdot \frac{\nu_{-2}}{\nu_{-2}} \\
\nu_2 &= \log(\nu_1) & \dot{\nu}_2 &= \frac{\dot{\nu}_1}{\nu_1} \\
\nu_3 &= \sqrt{\nu_0} & \dot{\nu}_3 &= 0.5 \cdot \frac{\dot{\nu}_0}{\nu_3} \\
\nu_4 &= \nu_{-1} \cdot \nu_3 & \dot{\nu}_4 &= \nu_{-1} \cdot \nu_3 + \nu_{-1} \cdot \frac{\dot{\nu}_3}{\nu_3} \\
\nu_5 &= \frac{\nu_2}{\nu_4} & \dot{\nu}_5 &= \frac{\dot{\nu}_2}{\nu_4} - \nu_5 \cdot \frac{\dot{\nu}_4}{\nu_4} \\
\nu_6 &= 0.5 \cdot \nu_4 & \dot{\nu}_6 &= 0.5 \cdot \dot{\nu}_4 \\
\nu_7 &= \nu_5 + \nu_6 & \dot{\nu}_7 &= \dot{\nu}_5 + \dot{\nu}_6 \\
\nu_8 &= \nu_7 - \nu_4 & \dot{\nu}_8 &= \dot{\nu}_7 - \dot{\nu}_4 \\
\nu_9 &= \omega \cdot \nu_7 & \dot{\nu}_9 &= \omega \cdot \dot{\nu}_7 \\
\nu_{10} &= \omega \cdot \nu_8 & \dot{\nu}_{10} &= \omega \cdot \dot{\nu}_8 \\
\nu_{11} &= \Phi(\nu_9) & \dot{\nu}_{11} &= \phi(\nu_9) \cdot \dot{\nu}_9 \\
\nu_{12} &= \Phi(\nu_{10}) & \dot{\nu}_{12} &= \phi(\nu_{10}) \cdot \dot{\nu}_{10} \\
\nu_{13} &= \nu_{-3} \cdot \nu_{11} & \dot{\nu}_{13} &= \nu_{-3} \cdot \nu_{11} + \nu_{-3} \cdot \dot{\nu}_{11} \\
\nu_{14} &= \nu_{-2} \cdot \nu_{12} & \dot{\nu}_{14} &= \nu_{-2} \cdot \nu_{12} + \nu_{-2} \cdot \dot{\nu}_{12} \\
\nu_{15} &= \nu_{13} - \nu_{14} & \dot{\nu}_{15} &= \dot{\nu}_{13} - \dot{\nu}_{14} \\
\nu_{16} &= \omega \cdot \nu_{15} & \dot{\nu}_{16} &= \omega \cdot \dot{\nu}_{15} \\
\gamma_1 &= \nu_{16} \\
\dot{\gamma}_1 &= \dot{\nu}_{16}
\end{align*} \]
Reverse Mode of AD calculates adjoints

- 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

\[
\bar{v}_j + = \bar{v}_i \cdot \frac{\partial}{\partial v_j} f_i(u_i).
\]

- In other words, for each arguments of $f_i$ the partial derivative is derived.
Reverse Mode of AD calculates adjoints 2/2

Reverse Mode or Adjoint Mode of AD

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

\[
\begin{align*}
    v_{i-n} &= x_i & i &= 1, \ldots, n \\
    v_i &= f_i(v_{j \prec i}) & i &= 1, \ldots, l \\
    y_{m-i} &= v_{l-i} & i &= m-1, \ldots, 0 \\
    \tilde{v}_i &= \tilde{y}_i & i &= 0, \ldots, m-1 \\
    \tilde{u}_i &= \tilde{f}_i(u_i, \tilde{v}_i) & i &= l, \ldots, 1 \\
    \tilde{x}_i &= \tilde{v}_i & i &= n, \ldots, 1.
\end{align*}
\]

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

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

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

- Computational effort is approx. 4 function evaluations of \( F \).
Black formula Reverse Mode evaluation procedure ...

\[
\begin{align*}
\nu_{-3} &= x_1 = F \\
\nu_{-2} &= x_2 = K \\
\nu_{-1} &= x_3 = \sigma \\
\nu_0 &= x_4 = \tau \\
\nu_1 &= \nu_{-3}/\nu_{-2} \\
\nu_2 &= \log(\nu_1) \\
\nu_3 &= \sqrt{\nu_0} \\
\nu_4 &= \nu_{-1} \cdot \nu_3 \\
\nu_5 &= \nu_2/\nu_4 \\
\nu_6 &= 0.5 \cdot \nu_4 \\
\nu_7 &= \nu_5 + \nu_6 \\
\nu_8 &= \nu_7 - \nu_4 \\
\nu_9 &= \omega \cdot \nu_7 \\
\nu_{10} &= \omega \cdot \nu_8 \\
\nu_{11} &= \Phi(\nu_9) \\
\nu_{12} &= \Phi(\nu_{10}) \\
\nu_{13} &= \nu_{-3} \cdot \nu_{11} \\
\nu_{14} &= \nu_{-2} \cdot \nu_{12} \\
\nu_{15} &= \nu_{13} - \nu_{14} \\
\nu_{16} &= \omega \cdot \nu_{15} \\
\bar{y}_1 &= \nu_{16} \\
\bar{\nu}_{16} &= \bar{y}_1 = 1
\end{align*}
\]
Black formula Reverse Mode evaluation procedure ... II

: 

\[ y_1 = v_{16} \]
\[ \bar{v}_{16} = \bar{y}_1 = 1 \]
\[ \bar{v}_{15} = \omega \cdot \bar{v}_{16} \]
\[ \bar{v}_{13} = \bar{v}_{15}; \quad \bar{v}_{14} = (-1) \cdot \bar{v}_{15} \]
\[ \bar{v}_{-2} = v_{12} \cdot \bar{v}_{14}; \quad \bar{v}_{12} = v_{-2} \cdot \bar{v}_{14} \]
\[ \bar{v}_{-3} = v_{11} \cdot \bar{v}_{13}; \quad \bar{v}_{11} = v_{-3} \cdot \bar{v}_{13} \]
\[ \bar{v}_{10} = \phi(v_{10}) \cdot \bar{v}_{12} \]
\[ \bar{v}_9 = \phi(v_9) \cdot \bar{v}_{11} \]
\[ \bar{v}_7 = \omega \cdot \bar{v}_9 \]
\[ \bar{v}_7 = \bar{v}_8; \quad \bar{v}_4 = (-1) \cdot \bar{v}_8 \]
\[ \bar{v}_5 = \bar{v}_7; \quad \bar{v}_6 = \bar{v}_7 \]
\[ \bar{v}_4 = 0.5 \cdot \bar{v}_6 \]
\[ \bar{v}_2 = \bar{v}_5/v_4; \quad \bar{v}_4 = (-1) \cdot v_5 \cdot \bar{v}_5/v_4 \]
\[ \bar{v}_{-1} = v_3 \cdot \bar{v}_4; \quad \bar{v}_3 = v_{-1} \cdot \bar{v}_4 \]
\[ \bar{v}_0 = 0.5 \cdot \bar{v}_3/v_3 \]
\[ \bar{v}_1 = \bar{v}_2/v_1 \]
\[ \bar{v}_{-3} = \bar{v}_{1}/v_{-2}; \quad \bar{v}_{-2} = (-1) \cdot v_1 \cdot \bar{v}_1/v_{-2} \]

\[ \bar{\tau} = \bar{x}_4 = \bar{v}_0 \]
\[ \bar{\sigma} = \bar{x}_3 = \bar{v}_{-1} \]
\[ \bar{K} = \bar{x}_2 = \bar{v}_{-2} \]
\[ \bar{F} = \bar{x}_1 = \bar{v}_{-3} \]
We summarise 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.
- Computational effort can be improved by AD vector mode.
- Reverse Mode memory consumption can be managed via checkpointing techniques.

**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 challenge for implementations.
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

There are also tools for Python and other languages:

More details at autodiff.org

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 is quite some literature on AD and its application in finance

Standard textbook on AD:

  SIAM, 2008

Recent practitioner’s textbook:

  SIAM, 2012

One of the first and influential papers for AD application in finance:

  *Risk*, January 2006
Part VIII

Wrap-up
Outline
What was this lecture about?

Interbank swap deal example

Trade details (fixed rate, notional, etc.)

- Bank A pays 3% on 100mm EUR
- Start date: Oct 31, 2019
- End date: Oct 31, 2039

(annually, 30/360 day count, modified following, Target calendar)

Date calculations

- Pays 6-months Euribor floating rate on 100mm EUR
- Start date: Oct 31, 2019
- End date: Oct 31, 2039

(semi-annually, act/360 day count, modified following, Target calendar)

Market conventions

Optionalities

Bank A may decide to early terminate deal in 10, 11, 12,.. years
F. Ametrano and M. Bianchetti.
Everything you always wanted to know about Multiple Interest Rate Curve Bootstrapping but were afraid to ask (April 2, 2013).

L. Andersen and V. Piterbarg.
*Interest rate modelling, volume I to III.*

D. Bang.
Local-stochastic volatility for vanilla modeling.

M. Beinker and H. Plank.
New volatility conventions in negative interest environment.
D. Brigo and F. Mercurio.

*Interest Rate Models - Theory and Practice.*


D. Duffy.

*Finite Difference Methods in Financial Engineering.*


M. Fujii, Y. Shimada, and A. Takahashi.

Collateral posting and choice of collateral currency - implications for derivative pricing and risk management (may 8, 2010).


M. Giles and P. Glasserman.

*Smoking adjoints: fast monte carlo greeks.*

*Risk,* January 2006.

P. Glasserman.

*Monte Carlo Methods in Financial Engineering.*

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

P. Hagan, D. Kumar, A. Lesniewski, and D. Woodward.  
Managing smile risk.  

P. Hagan and G. West.  
Interpolation methods for curve construction.  

M. Henrard.  
Interest rate instruments and market conventions guide 2.0.  
Open Gamma Quantitative Research, 2013.

M. Henrard.  
A quant perspective on ibor fallback proposals.  
J. Hull and A. White. 
Pricing interest-rate-derivative securities. 

Y. Iwashita. 
Piecewise polynomial interpolations. 
OpenGamma Quantitative Research, 2013.

U. Naumann. 
The Art of Differentiating Computer Programs: An Introduction to Algorithmic Differentiation. 

V. Piterbarg. 
Funding beyond discounting: collateral agreements and derivatives pricing. 
Asia Risk, pages 97–102, February 2010.
R. Rebonato.  
*Volatility and Correlation.*  

S. Shreve.  
*Stochastic Calculus for Finance II - Continuous-Time Models.*  
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