

# Interest Rate Modelling and Derivative Pricing

Sebastian Schlenkrich

HU Berlin, Department of Mathematics

Summer term, 2023

## 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 (1/2)

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

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

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

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

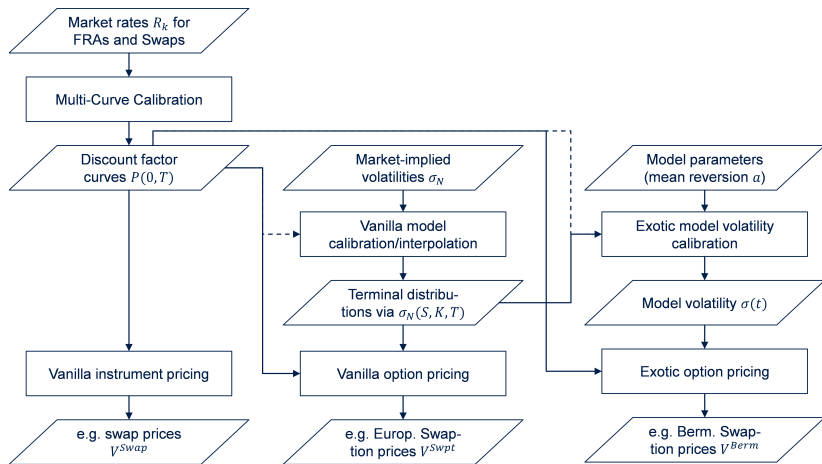
## Market risk calculation relies on accurate sensitivities (2/2)

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

- ▶ 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

- ▶ 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 (1/2)

### 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 (2/2)

### Parallel Delta and Gamma

Parallel Delta and Gamma represent sensitivities w.r.t. simultaneous shifts of all market rates of a yield curve. With  $\mathbf{1} = [1, \dots, 1]^\top$  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 (1/2)

## Bucketed ATM Normal Volatility Vega

Denote  $\bar{\sigma} = \left[ \sigma_N^{k,l} \right]$  the matrix of market-implied At-the-money normal volatilities for expiries  $k = 1, \dots, q$  and swap terms  $l = 1, \dots, r$ .

Bucketed ATM Normal Volatility Vega of an instrument with model price  $V = V(\bar{\sigma})$  is specified as

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

# Vega is the sensitivity w.r.t. changes in market volatilities (2/2)

## 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

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

- ▶ Volatility smile sensitivities are often specified in terms of Vanilla model parameter sensitivities.
- ▶ For example, in SABR model, we can calculate sensitivities with respect to  $\alpha$ ,  $\beta$ ,  $\rho$  and  $\nu$ .

# Outline

Introduction to Sensitivity Calculation

**Finite Difference Approximation for Sensitivities**

Differentiation and Calibration

A brief Introduction to Algorithmic Differentiation

## 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

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

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

- ▶ Simple to implement and calculate; only pricing function evaluation.
- ▶ Typically used for black-box pricing functions.



# We do a case study for European swaption Vega I

Recall pricing function

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

with

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

First, analyse Bachelier formula. We get

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

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

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

## 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)

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

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

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

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

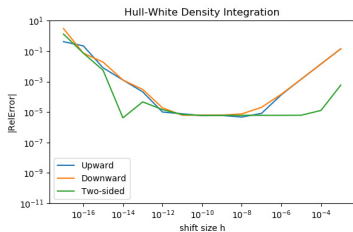
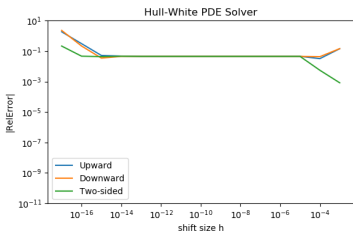
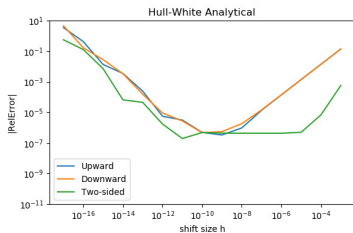
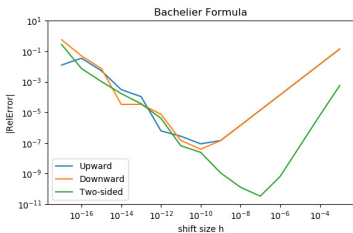
$$\text{Vega} = An(t) \cdot \Phi'(h) \cdot \sqrt{T - t}.$$

- ▶ 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?

II



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 (1/2)

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

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

## Derivative pricing usually involves model calibration (2/2)

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

- ▶ 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  $\mathcal{H} : \mathbb{R}^q \times \mathbb{R}^r \rightarrow \mathbb{R}^q$  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  $g : \mathcal{U} \rightarrow \mathbb{R}^q$  with

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

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.



# How does Implicit Function Theorem help for sensitivity calculation? (1/4)

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

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

- ▶ For example, for swap rate helpers, model-implied par swap rate becomes

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

## How does Implicit Function Theorem help for sensitivity calculation? (2/4)

Suppose 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, by Implicit Function Theorem, there exists a function

$$z = z(R)$$

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

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

## How does Implicit Function Theorem help for sensitivity calculation? (3/4)

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

From reformulated calibration helpers we get

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

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

Consequently

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

# How does Implicit Function Theorem help for sensitivity calculation? (4/4)

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 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 also adapt Jacobian method to Vega calculation (1/3)

Bermudan swaption is determined via mapping

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

Assign volatility calibration helpers

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

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

## We can also adapt Jacobian method to Vega calculation (2/3)

Implicit Function Theorem yields

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

- ▶  $\frac{d}{d\sigma} \text{Model}[\sigma]$  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.

## We can also adapt Jacobian method to Vega calculation (3/3)

Bermudan Vega becomes

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



# 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

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

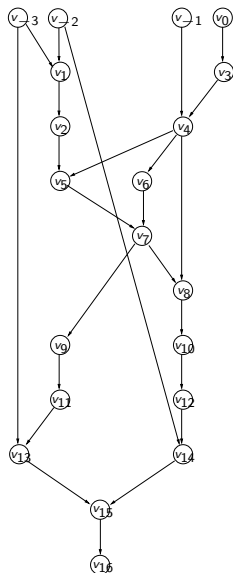
$$\text{with } d_{1,2} = \frac{\log(F/K)}{\sigma\sqrt{\tau}} \pm \frac{\sigma\sqrt{\tau}}{2}$$

- ▶ Inputs  $F, K, \sigma, \tau$
- ▶ Discrete parameter  $\omega \in \{-1, 1\}$
- ▶ Output  $\text{Black}(\cdot)$

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

# Alternative representation is Directed Acyclic Graph (DAG)

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



# Evaluation Procedure can be formalized to make it more tractable

## Definition (Evaluation Procedure)

Suppose  $F : \mathbb{R}^n \rightarrow \mathbb{R}^m$  and  $f_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^{m_i}$ . 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

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

then we call this sequence of operations an evaluation procedure of  $F$  with elementary operations  $f_i$ . We assume differentiability of all elementary operations  $f_i$  ( $i = 1, \dots, l$ ). 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 (1/2)

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

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

### Forward Mode or Tangent Mode of AD

Use abbreviations  $\dot{u}_i = (\dot{v}_j)_{j < i}$  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

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

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

## Forward mode of AD calculates tangents (2/2)

- ▶ 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...

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



## Reverse Mode of AD calculates adjoints (1/3)

- ▶ 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 (2/3)

### Reverse Mode or Adjoint Mode of AD

Denoting  $\bar{u}_i = (\bar{v}_j)_{j < i} \in \mathbb{R}^{n_i}$  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

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

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  $y_i$ .

## Reverse Mode of AD calculates adjoints (3/3)

- ▶ 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

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

- ▶ Computational effort is approx. 4 function evaluations of  $F$ .

# Black formula Reverse Mode evaluation procedure ... I

$$v_{-3} = x_1 = F$$

$$v_{-2} = x_2 = K$$

$$v_{-1} = x_3 = \sigma$$

$$v_0 = x_4 = \tau$$

---

$$v_1 = v_{-3}/v_{-2}$$

$$v_2 = \log(v_1)$$

$$v_3 = \sqrt{v_0}$$

$$v_4 = v_{-1} \cdot v_3$$

$$v_5 = v_2/v_4$$

$$v_6 = 0.5 \cdot v_4$$

$$v_7 = v_5 + v_6$$

$$v_8 = v_7 - v_4$$

$$v_9 = \omega \cdot v_7$$

$$v_{10} = \omega \cdot v_8$$

$$v_{11} = \Phi(v_9)$$

$$v_{12} = \Phi(v_{10})$$

$$v_{13} = v_{-3} \cdot v_{11}$$

$$v_{14} = v_{-2} \cdot v_{12}$$

$$v_{15} = v_{13} - v_{14}$$

$$v_{16} = \omega \cdot v_{15}$$

---

$$y_1 = v_{16}$$

$$\bar{y}_{16} = \bar{y}_1 = 1$$

---

⋮

# Black formula Reverse Mode evaluation procedure ... II

⋮

---


$$y_1 = v_{16}$$

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


---

$$\bar{v}_{15} += \omega \cdot \bar{v}_{16}$$

$$\bar{v}_{13} += \bar{v}_{15}; \quad \bar{v}_{14} += (-1) \cdot \bar{v}_{15}$$

$$\bar{v}_{-2} += v_{12} \cdot \bar{v}_{14}; \quad \bar{v}_{12} += v_{-2} \cdot \bar{v}_{14}$$

$$\bar{v}_{-3} += v_{11} \cdot \bar{v}_{13}; \quad \bar{v}_{11} += v_{-3} \cdot \bar{v}_{13}$$

$$\bar{v}_{10} += \phi(v_{10}) \cdot \bar{v}_{12}$$

$$\bar{v}_9 += \phi(v_9) \cdot \bar{v}_{11}$$

$$\bar{v}_8 += \omega \cdot \bar{v}_{10}$$

$$\bar{v}_7 += \omega \cdot \bar{v}_9$$

$$\bar{v}_7 += \bar{v}_8; \quad \bar{v}_4 += (-1) \cdot \bar{v}_8$$

$$\bar{v}_5 += \bar{v}_7; \quad \bar{v}_6 += \bar{v}_7$$

$$\bar{v}_4 += 0.5 \cdot \bar{v}_6$$

$$\bar{v}_2 += \bar{v}_5/v_4; \quad \bar{v}_4 += (-1) \cdot v_5 \cdot \bar{v}_5/v_4$$

$$\bar{v}_{-1} += v_3 \cdot \bar{v}_4; \quad \bar{v}_3 += v_{-1} \cdot \bar{v}_4$$

$$\bar{v}_0 += 0.5 \cdot \bar{v}_3/v_3$$

$$\bar{v}_1 += \bar{v}_2/v_1$$

$$\bar{v}_{-3} += \bar{v}_1/v_{-2}; \quad \bar{v}_{-2} += (-1) \cdot v_1 \cdot \bar{v}_1/v_{-2}$$


---

$$\bar{r} = \bar{x}_4 = \bar{v}_0$$

$$\bar{\sigma} = \bar{x}_3 = \bar{v}_{-1}$$

$$\bar{K} = \bar{x}_2 = \bar{v}_{-2}$$

$$\bar{F} = \bar{x}_1 = \bar{v}_{-3}$$

# 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

### 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 languages:

More details at [autodiff.org](http://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 influential papers for AD application in finance:

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



# Part VIII

## Wrap-up

# Outline

# What was this lecture about?

## Interbank swap deal example

### Trade details (fixed rate, notional, etc.)

Pays 3% on 100mm EUR

Start date: Oct 30, 2020

End date: Oct 30, 2040

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

### Date calculations

### Market conventions



### Stochastic interest rates

Pays 6-months Euribor floating rate on 100mm EUR

Start date: Oct 30, 2020

End date: Oct 30, 2040

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

### Optionalities

Bank A may decide to early terminate deal in 10, 11, 12,.. years

# Outline

## References

# References I



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).

Available at SSRN: <http://ssrn.com/abstract=2219548> or <http://dx.doi.org/10.2139/ssrn.2219548>, 2013.



L. Andersen and V. Piterbarg.

*Interest rate modelling, volume I to III.*

Atlantic Financial Press, 2010.



D. Bang.

Local-stochastic volatility for vanilla modeling.

<https://ssrn.com/abstract=3171877>, 2018.



M. Beinker and H. Plank.

New volatility conventions in negative interest environment.

*d-fine Whitepaper, available at [www.d-fine.de](http://www.d-fine.de), December 2012.*



D. Brigo and F. Mercurio.

*Interest Rate Models - Theory and Practice.*

Springer-Verlag, 2007.

# References II



D. Duffy.

*Finite Difference Methods in Financial Engineering.*

Wiley Finance, 2006.



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

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

Available at SSRN: <https://ssrn.com/abstract=1601866>, May 2010.



M. Giles and P. Glasserman.

Smoking adjoints: fast monte carlo greeks.

*Risk*, January 2006.



P. Glasserman.

*Monte Carlo Methods in Financial Engineering.*

Springer, 2003.



A. Griewank and A. Walther.

*Evaluating derivatives: principles and techniques of algorithmic differentiation - 2nd ed.*

SIAM, 2008.

# References III



P. Hagan, D. Kumar, A. Lesniewski, and D. Woodward.  
Managing smile risk.  
*Wilmott magazine*, September 2002.



P. Hagan and G. West.  
Interpolation methods for curve construction.  
*Applied Mathematical Finance*, 13(2):89–128, 2006.



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.  
<https://ssrn.com/abstract=3226183>, 2018.



M. Henrard.  
A quant perspective on ibor fallback consultation results.  
<https://ssrn.com/abstract=3308766>, 2019.

# References IV



J. Hull and A. White.

Pricing interest-rate-derivative securities.

*The Review of Financial Studies*, 3:573–592, 1990.



Y. Iwashita.

Piecewise polynomial interpolations.

OpenGamma Quantitative Research, 2013.



A. Lyashenko and F. Mercurio.

Looking forward to backward-looking rates: A modeling framework for term rates replacing libor.

<https://ssrn.com/abstract=3330240>, 2019.



U. Naumann.

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

SIAM, 2012.



V. Piterbarg.

Funding beyond discounting: collateral agreements and derivatives pricing.

*Asia Risk*, pages 97–102, February 2010.



# References V



R. Rebonato.

*Volatility and Correlation.*

John Wiley & Sons, 2004.



S. Shreve.

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

Springer-Verlag, 2004.

# Contact

Dr. Sebastian Schlenkrich

Office: RUD25, R 1.201

Mail: [sebastian.schlenkrich@hu-berlin.de](mailto:sebastian.schlenkrich@hu-berlin.de)

