Design Patterns for Algorithmic Differentiation

Dr. Sebastian Schlenkrich
QuantLib Workshop, November 2013

Joint work with Jan Riehme, RWTH Aachen

Agenda

1. Sensitivities of Exotic Derivatives

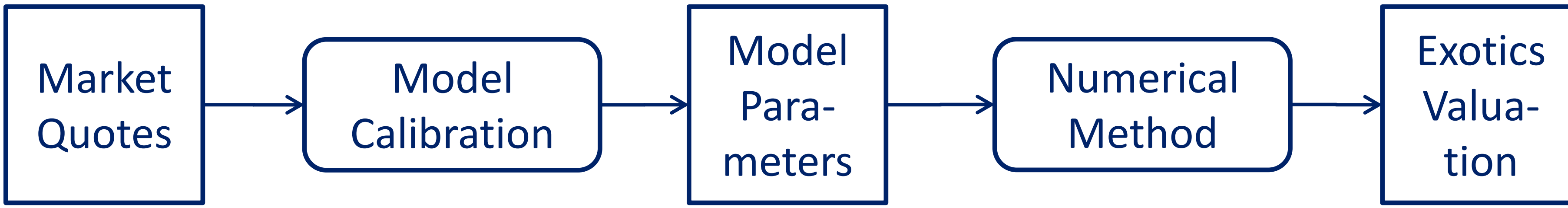
2. Algorithmic Differentiation (AD) at a Glance

3. Incorporation of AD Methodologies into Financial Libraries

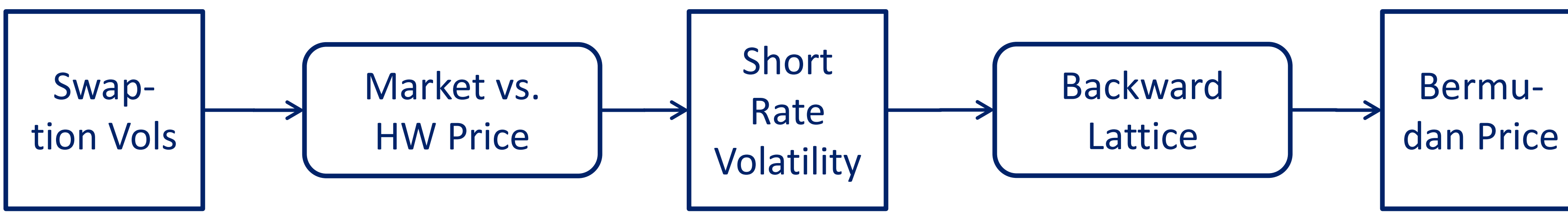
4. Proof of Concept for Bermudan Swaption Vega in QuantLib

Sensitivities of Exotic Derivatives

Generic Valuation Process for Exotics

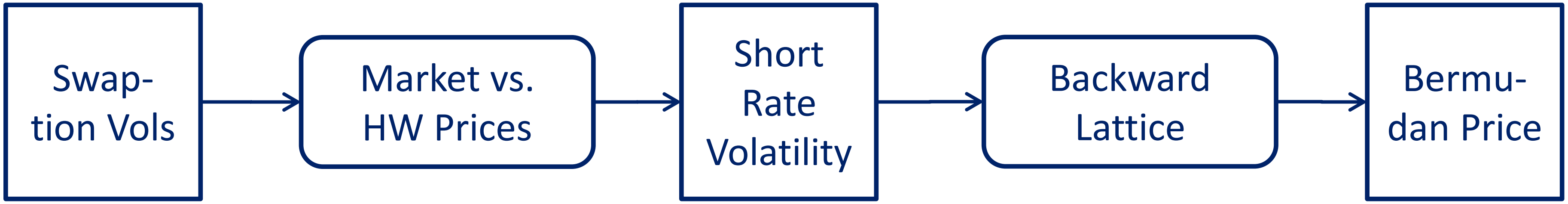


Example: Bermudan Swaptions with Hull White Model



Notations and Mappings

Example: Bermudan Swaptions with Hull White Model

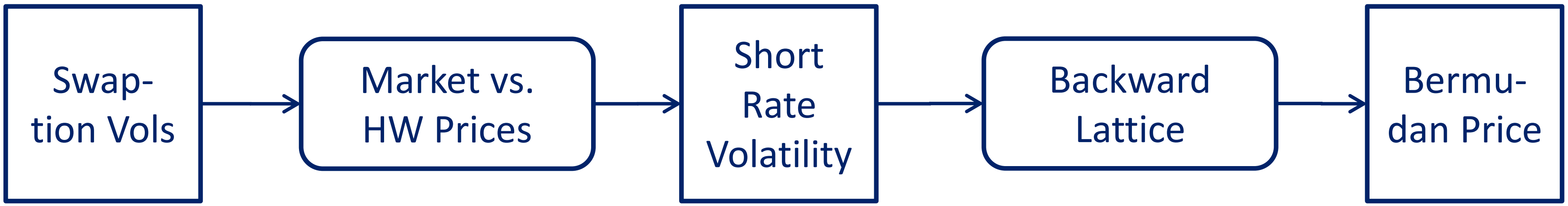


σ_{B76} $\quad$ $Market(\sigma_{B76})$ $Model(\sigma_{HW})$ $\quad$ σ_{HW} $\quad$ $Exotic(\sigma_{HW})$ $\quad$ V

- Vector of Swaption volatilities σ_{B76}
- Vector of Hull White short rate volatility term structure σ_{HW}
- Vector functions for Swaption prices $Market(\sigma_{B76})$ and $Model(\sigma_{HW})$ using Black'76 and Hull White analytical model formulas
- Exotics valuation function $\mathrm{V} = Exotic(\sigma_{HW})$ based on model parameters

Exotics Sensitivity Evaluation

- Assume invertability and differentiability of functions involved



$$\sigma_{B76} \xrightarrow{Market(\cdot)} Swpt \xrightarrow{Model^{-1}(\cdot)} \sigma_{HW} \xrightarrow{Exotic(\cdot)} V$$

$$V = Exotics\left(Model^{-1}\left(Market(\sigma_{B76})\right)\right)$$

$$\boldsymbol{\frac{dV}{d\sigma_{B76}} = Exotic'(\sigma_{HW}) \cdot Model'(\sigma_{HW})^{-1} \cdot Market'(\sigma_{B76})}$$

Sensitivities Involved

$$\frac{dV}{d\sigma_{B76}} = Exotic'(\sigma_{HW}) \cdot Model'(\sigma_{HW})^{-1} \cdot Market'(\sigma_{B76})$$

- Analytic Vega formula

- Analytic expression available, but tedious
- More complex models than Hull White may not allow analytic derivative formulas

- Numerical method in general does not exhibit analytic derivative
- Remedies: - Finite difference approximations
  - **Algorithmic differentiation**

Algorithmic Differentiation at a Glance

Algorithmic Differentiation (AD)

- Principles and techniques to augment computer models
- Sensitivities of output variables with respect to inputs of the model
- Numerical values rather than symbolic expressions
- Sensitivities exact up to machine precision (no rounding/cancellation errors)
- Apply chain rule of differentiation to operations like „`+`“, „`*`“, „`exp()`“, …

Example: Black'76 Vega of ATM Option

$$V = F\left[2N\left(\sigma\sqrt{T}/2\right) - 1\right], \qquad dV/d\sigma = F\phi\left(\sigma\sqrt{T}/2\right)\sqrt{T}$$

Single Assignment Code of Elementary Operations

	Original Computer Model	Augmented Computer Model
Initialisation	F, σ and T	$\dot{F} = 0, \dot{\boldsymbol{\sigma}} = \mathbf{1}$ and $\dot{T} = 0$
Evaluation	$v_1 = \sqrt{T}$	$\dot{v}_1 = 1/(2v_1)$
	$v_2 = \sigma \cdot v_1$	$\dot{v}_2 = \dot{\sigma} \cdot v_1 + \sigma \cdot \dot{v}_1$
	$v_3 = v_2/2$	$\dot{v}_3 = \dot{v}_2/2$
	$v_4 = N(v_3)$	$\dot{v}_4 = \phi(v_3) \cdot \dot{v}_3$
	$v_5 = 2 \cdot v_4$	$\dot{v}_5 = 2 \cdot \dot{v}_4$
	$v_6 = v_5 - 1$	$\dot{v}_6 = \dot{v}_5$
	$v_7 = F \cdot v_6$	$\dot{v}_7 = \dot{F} \cdot v_6 + F \cdot \dot{v}_6$
Result	$\boldsymbol{V = v_7}$	$\boldsymbol{\dot{V} = \dot{v}_7}$

Implementation and Tools

Methodologies

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

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

Some Tools for C++

Source Code Transformation: ADIC2, dcc, TAPENADE

Operator Overloading: ADOL-C, dco, ADMB/AUTODIF

Some References for Automatic Differentiation

Community Website

www.autodiff.org

Standard Text Book

A. Griewank, A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, 2nd Edition. 2008

Recent Practitioner's Text Book

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

Incorporation of AD into Financial Libraries

Practical Considerations for AD in Software Packages

- Source transformation and overloading result in new AD-enabled model
- AD-enabled model needs to be maintained consistently in software development cycle besides original model
  - E.g. by re-creation of AD model after each original model update
- AD model usually does not implement the interface of the original model
  - Sensitivity evaluation needs to be wrapped appropriately

Handling

Templatisation and Operator Overloading

Object Adapter Design Pattern

Example: Bermudan Swaption with Hull White Model

Hull White Model Valuation

- European Swaptions as European Coupon Bond Options (CBO)
- Bermudan Swaptions as Bermudan CBO

Hull White Model Vegas

- d [Europ. CBO Price] / d [short rate volatility] = $Model'(\sigma_{HW})$
- d [Berm. CBO Price] / d [short rate volatility] = $Exotic'(\sigma_{HW})$

Operator Overloading AD Tool ADTAGEO*

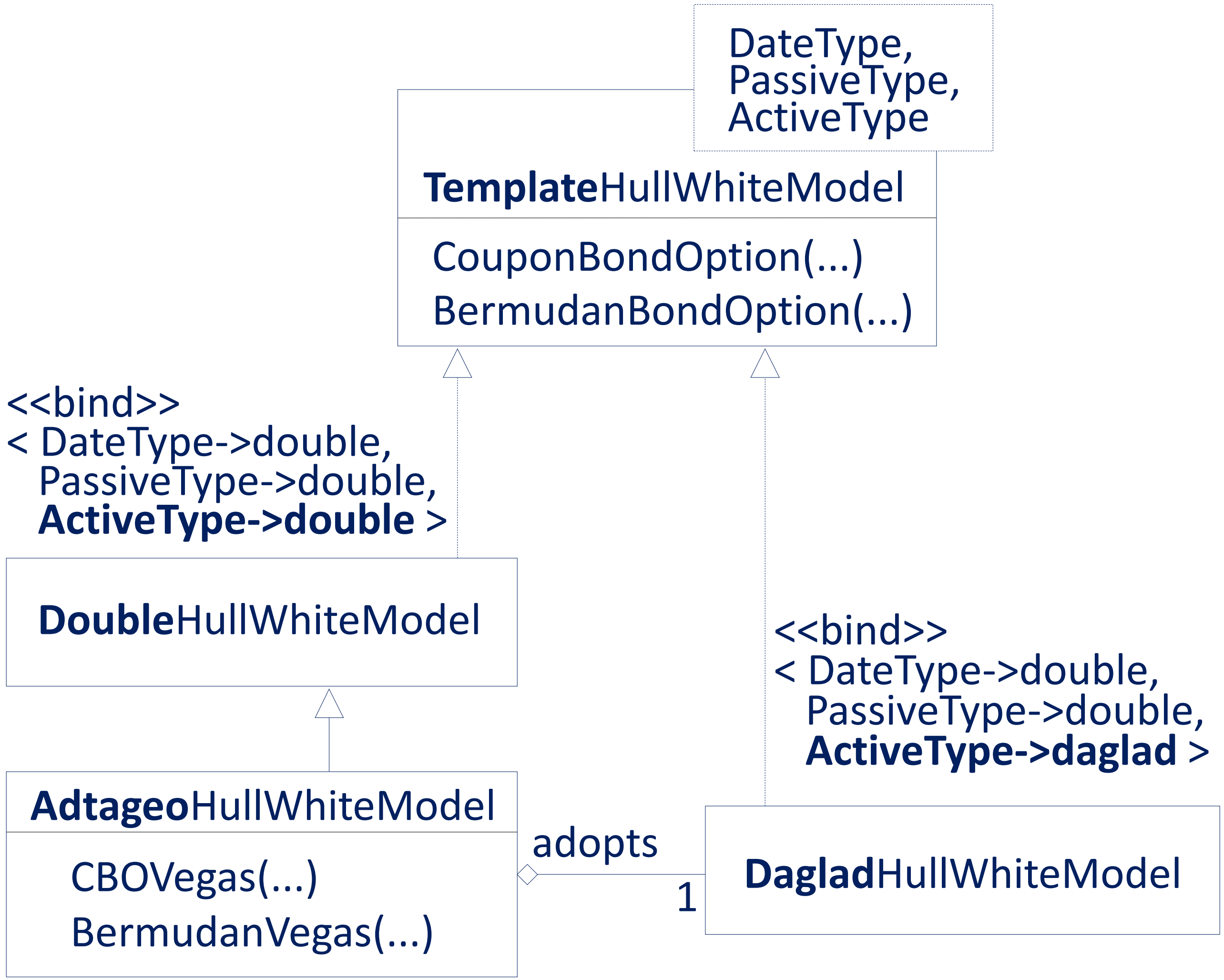
- **A**lgorithmic **D**ifferentiation **T**rough **A**utomatic **G**raph **E**limination **O**rdering
- Sensitivity aware user defined data type **`daglad`**
- Sensitivity dy/dx for $y = f(x)$ via % operator, that is **`dydx = y % x`**

* J. Riehme and A. Griewank, *Algorithmic differentiation through automatic graph elimination ordering (adtageo)*, in U. Naumann, O. Schenk, H. D. Simon and S. Toledo (eds), Combinatorial Scientifc Computing, number 09061 in Dagstuhl Seminar Proceedings. 2009.

Pure Template Based Model Definition

```
template<class DateType, class PassiveType, class ActiveType>
class TemplateHullWhiteModel {
    std::vector<DateType> volaDates;
    std::vector<ActiveType> volaValues;
    PassiveType meanReversion;
    ...
    virtual ActiveType CouponBondOption(...);
    virtual ActiveType BermudanBondOption(...);
};
```

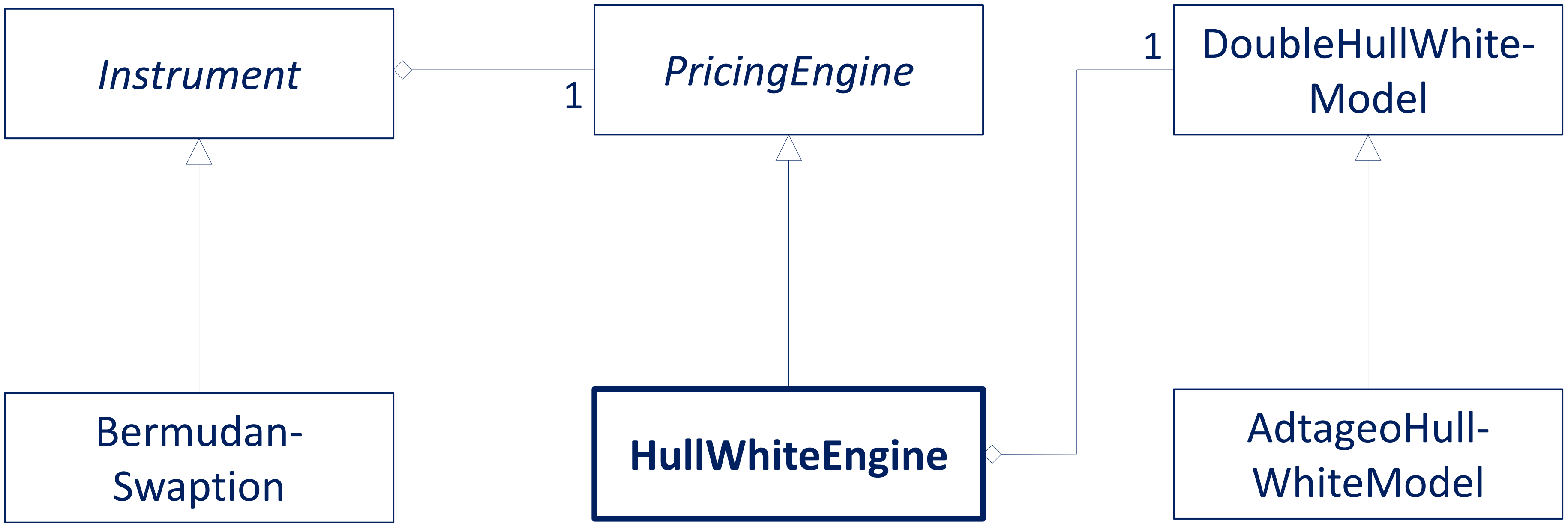
Object Adapter Design Pattern



Algorithmic Differentiation Enabled Adapter Class

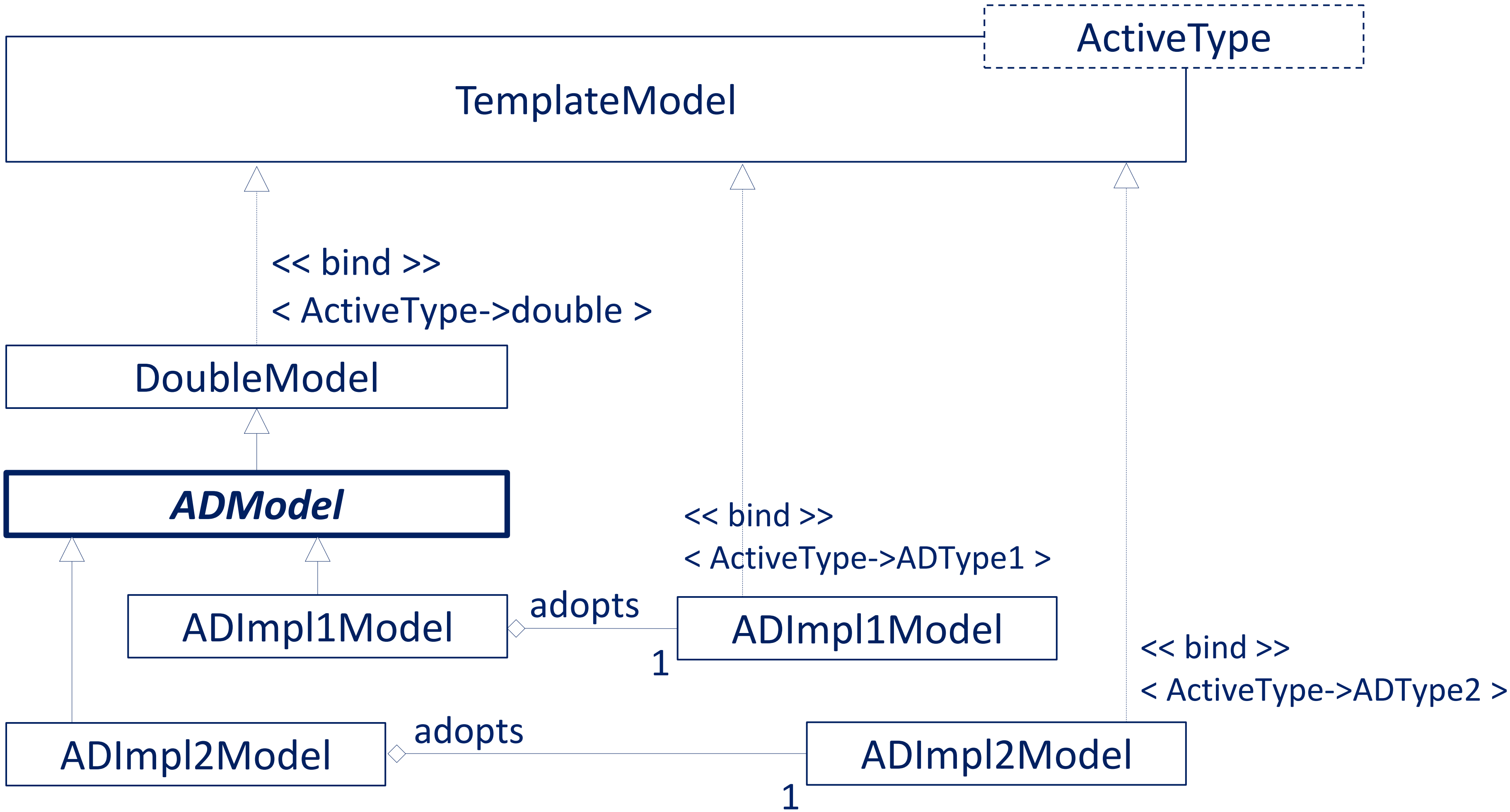
```
class AdtageoHullWhiteModel : public DoubleHullWhiteModel {
    DagladHullWhiteModel *aModel;
    std::vector<double> bermudanVegas;
    ...
    virtual double BermudanBondOption(...) {
        daglad res = aModel->BermudanBondOption(...);
        for (size t i=0; i<bermudanVegas.size(); ++i)
            bermudanVegas[i] = res % aModel->volaValues[i];
        return res.val();
    }
    virtual std::vector<double> BermudanVegas() {
        return bermudanVegas;
    } ...
};
```

Flexible Incorporation into QuantLib Framework



- Map instrument to Bermudan CBO
- Calibrate Hull White Model
- Evaluate NPV
- **Try downcast and request Vegas**

Generalization for Several AD Tool Implementations



Proof of Concept for Bermudan Swaption Vega in QuantLib

QuantLib Object Setup in Excel

HullWhiteModel Error	HullWhiteModel#0007
ObjectID	HullWhiteModel
DiscountCurve	6M-EUR-Swap-Curve#0000
MeanReversion	0.084
VolaTimes	0
VolaValues	0.01
Permanent	
Trigger	
OverWrite	

AD HullWhiteModel Error	ADHullWhiteModel#0007
ObjectID	ADHullWhiteModel
DiscountCurve	6M-EUR-Swap-Curve#0000
MeanReversion	0.084
VolaTimes	0
VolaValues	0.008749649
Permanent	
Trigger	
OverWrite	

BondOptionEngine Error	dOptionEngineSwaption#0008
ObjectID	BondOptionEngineSwaption
HullWhiteModel	ADHullWhiteModel#0007
Dimension	1001
GridRadius	0.3
BermudanTolerance	1.00E-04
SwaptionProperties	SwaptionProperties#0007
CalibrationTolerance	1.00E-10
Permanent	
Trigger	
OverWrite	

SetPricingEngine	TRUE
NPV	0.028766898
ErrorEstimate	6.54674E-06
Bermudan Vega	19.20%
EstimateAccuracy	TRUE
TRUE	TRUE

Detailed QuantLib Valuation Results in Excel

Bermudan NPV: 2.877%

Vega as 1 unit shift sensitivity

ExerciseDates	B76Vola	B76Prices	B76Vega	VolaTimes	VolaValues	Europ.Analyt.	Europ.Num.	BermVegas
30.11.2011	27.79%	1.727%	9.77%	1.0	1.311%	1.727%	1.727%	**3.29%**
30.11.2012	26.13%	1.885%	12.16%	2.0	1.233%	1.885%	1.885%	**3.49%**
29.11.2013	24.27%	1.724%	12.84%	3.0	1.103%	1.724%	1.724%	**2.69%**
28.11.2014	22.63%	1.507%	12.60%	4.0	0.997%	1.507%	1.507%	**2.04%**
30.11.2015	21.42%	1.309%	11.80%	5.0	0.950%	1.309%	1.309%	**1.67%**
30.11.2016	20.83%	1.150%	10.62%	6.0	0.994%	1.150%	1.150%	**1.64%**
30.11.2017	20.12%	0.944%	9.02%	7.0	0.905%	0.944%	0.944%	**1.36%**
30.11.2018	19.75%	0.743%	7.14%	8.0	0.930%	0.743%	0.743%	**1.27%**
29.11.2019	19.16%	0.504%	4.97%	9.0	0.821%	0.504%	0.504%	**0.97%**
30.11.2020	18.96%	0.267%	2.59%	10.0	0.876%	0.267%	0.267%	**0.78%**
Sum								19.20%

A flat 1% shift in Swaption volatilities yields a 0.192% Bermudan NPV shift

Conclusions

Conclusions

- Market sensitivities for Exotics can be evaluated by differentiating calibration and Exotics instrument model pricers
- Algorithmic Differentiation (AD) methodologies yield accurate sensitivites
- Model templatisation and object adapter design patterns are flexible concepts to incorporate Operator Overloading AD methodologies

QuantLib should make use of template-based model implementations

Your Contacts at d-fine

Dr Sebastian Schlenkrich
Manager
Mobile +49 162 263-1525
E-Mail: Sebastian.Schlenkrich@d-fine.de

Dr Mark Beinker
Head of Business Unit Applied Financial Engineering
Partner
Tel +49 69 90737-305
E-Mail: Mark.Beinker@d-fine.de

d-fine GmbH
Opernplatz 2
60313 Frankfurt am Main
Germany
Tel +49 69 90737-0
www.d-fine.de

d-fine Ltd
6-7 Queen St
London
EC4N 1SP
Tel +44 20 7776-1000
www.d-fine.co.uk