

Equity Derivative Structuring Case Study

OVERVIEW	1
PRODUCT DEFINITION	1
SAMPLE PRODUCT.....	1
MLFi DEFINITION	2
SIMULATION	5
SIMULATION OF CONTRACT'S EXECUTION (PERFORMANCE ANALYSIS)	5
SIMULATION OF FUTURE PRICE.....	7
SIMULATION AUDIT	8
PRICING.....	8
PRICE, GREEKS, PAYOFF DISTRIBUTION	8
SENSITIVITY ANALYSIS.....	9
EVENT PLANNING	11
EVENT DETECTION.....	11
CONTRACT EXPLORER	12
FROM PROTOTYPING TO PRODUCTION	12
PRODUCT DESIGN, USER INTERFACE, AND PERSISTENCE.....	12
RAPID PROTOTYPING AND STAGED DEPLOYMENT OF NEW VALUATION ALGORITHMS	13
REPORTING	13
CONCLUSION.....	14

Overview

This document illustrates the process of structuring a new financial product with LexiFi. The case study is based on a multi-period, multi-underlying equity derivative, representative of the products marketed in European retail markets.

Topics covered include:

- Product definition
- Simulation
- Pricing
- Event planning
- Transition from prototyping to production

Product Definition

Sample Product

The sample contract is based on a basket of twenty assets and pays 85% of the basket's initial value plus the highest positive average return calculated on ten annual observation dates.

Each year, the basket's value is calculated using the following rules:

- As soon as an asset becomes the best performer on a given year, that year's performance, subject to a performance boost applicable in each of the first seven annual observation dates (see below), is used in the calculation of the average basket's performance not only for the given year but also

for the following years. The contribution of a once best performing asset to the value of the basket is effectively “frozen” on the date the asset becomes the best performer. The basket’s value is therefore insensitive to future changes in the value of a once best performing asset.

- On each of the first seven annual observation dates, if the best performing asset returns less than 100%, the performance is set to 100%.
- The basket’s value on each annual observation date is equal to the arithmetic average of individual asset returns.
- The contract’s payoff is equal to 85% of the basket’s initial value plus the maximum of the ten average basket returns or zero if the maximum is negative.

The following table illustrates the contract’s mechanics:

Table 1
Asset Spot Prices (Scenario 0) and Illustration of Sample Contract’s Mechanics

	2003- 11-14	2004- 11-15	2005- 11-14	2006- 11-14	2007- 11-14	2008- 11-14	2009- 11-16	2010- 11-15	2011- 11-14	2012- 11-14	2013- 11-06
Asset 1	100	106	113	156	189	195	268	268	268	268	268
Asset 2	100	95	99	105	169	115	89	95	102	198	202
Asset 3	100	156	198	156	148	196	210	165	189	165	143
Asset 4	100	159	147	218	213	157	214	226	245	245	245
Asset 5	100	98	97	101	103	159	125	265	265	265	265
Asset 6	100	110	126	123	157	143	121	107	99	101	107
Asset 7	100	114	107	113	154	134	147	168	159	148	156
Asset 8	100	173 hence 200	200	200	200	200	200	200	200	200	200
Asset 9	100	105	198	123	159	143	156	123	156	189	201
Asset 10	100	159	199 hence 200	200	200	200	200	200	200	200	200
Asset 11	100	150	195	242	268	268	268	268	268	268	268
Asset 12	100	121	156	189	168	198	201	168	149	165	203
Asset 13	100	99	94	93	96	97	91	95	97	92	57
Asset 14	100	124	144	168	182	197	205	231	243	301	301
Asset 15	100	158	173	187	181	197	212	196	217	217	194
Asset 16	100	157	152	158	205	199 hence 200	200	200	200	200	200
Asset 17	100	109	115	126	134	146	157	163	178	164	164
Asset 18	100	113	135	154	174	196	157	176	198	176	137
Asset 19	100	154	192	243	243	243	243	243	243	243	243
Asset 20	100	134	164	199	221	198	203	198	233	222	237
Average		131.05	150.25	162.7	178.2	179.1	183.35	187.75	195.45	201.35	199.55

Maximum

The highlighted cells identify the contributions of best performing assets. The highest average basket return, identified by the yellow cell, is obtained on 2012-11-14. On the payment date, the holder of the contract receives:

$$85\% + \text{Max} (201.35/100 - 1, 0) = 186.35\%$$

MLFi Definition

New Product vs. Existing Product Template

The skills required to describe a product with LexiFi depend on the task at hand. Two situations may be distinguished:

- *Defining a new product.* Structurers define entirely new payoffs using the MLFi language and libraries of higher-level contract assembly components. Five to twenty lines of MLFi code generally suffice to capture the product’s logic. If required, LexiFi assists customers in designing new contracts.

- *Specifying the parameters of a pre-defined product template.* Once the core logic of the contract is written in MLFi, the definition is extended to enable non-technical users to instantiate the contract with parameters. Salespeople may specify simple parameters (e.g., a date to describe a maturity date, a float to describe a notional amount) or more complex parameters (e.g., an algebraic expression to describe a complex payoff).

Parametric MLFi product definitions are used to generate trade entry screens with Windows Forms, the application programming interface for writing graphical applications, delivered as part of the Microsoft .NET framework.

In practice, structurers progressively expand the library of high-level contract assembly components and create increasingly general product templates. Over time, a growing proportion of “new” products is defined by modifying the parameters of an existing structure instead of writing MLFi code.

For their part, salespeople access a catalog of product templates that defines the potential range of solutions that they may offer to customers. The sales team is able to autonomously adapt products in the catalog to meet customer needs.

Possible Parametric Representation

The table below presents a possible parametric representation of the sample product.

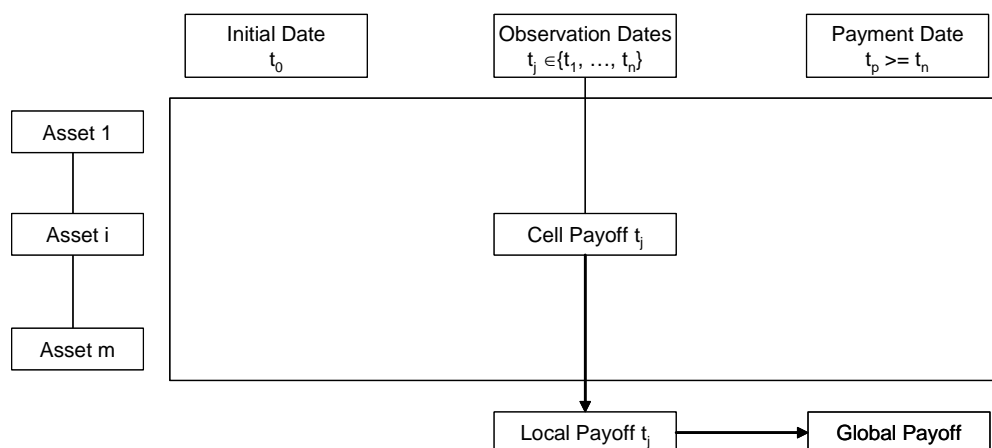
Table 2
Possible Parametric Representation of Sample Product

Parameter Name / MLFi Type / Description	Parameter Value
initial_date date Initial observation date. The values of underlyings on the initial date serve as denominators for calculating the performance of relevant assets on each annual observation date.	2003-11-14
Assets string list List of underlying assets.	["asset01"; "asset02"; "asset03"; "asset04"; "asset05"; "asset06"; "asset07"; "asset08"; "asset09"; "asset10"; "asset11"; "asset12"; "asset13"; "asset14"; "asset15"; "asset16"; "asset17"; "asset18"; "asset19"; "asset20"]
observation_dates date list Dates on which the values of underlyings are observed in order to calculate each asset's performance.	[2004-11-15; 2005-11-14; 2006-11-14; 2007-11-14; 2008-11-14; 2009-11-16; 2010-11-15; 2011-11-14; 2012-11-14; 2013-11-06]
number_withdrawn int * (int * int) list Specifies the number of assets withdrawn from the basket on observation date t_j . always 1 means that one asset is withdrawn on each observation date.	always 1
cell_payoffs string * (int * string) list Defines the function applied to the performance of the best performing asset on observation date t_j in order to determine the value used in the local payoff calculation. The first string is the default expression that characterizes such function, and the (int * string) list describes the observation date index from which a new expression starts to apply and the new expression. The list is empty when the default expression applies to all observation dates.	"max 2 CELL", [8, "CELL"]

<p>In the opposite column, "min 2 CELL" means that function $f: x \rightarrow \min 2 x$ is applied to the performance of the best performing asset from the first observable date t_1. The performance boost is abandoned from the eighth observation date, as specified by "8, CELL". CELL is the only primitive that may be used to defined cell payoff expressions.</p>	
<p>local_payoffs string * (int * string) list Defines the function applied (vertically) to cell payoffs on observation date t_j to calculate the local payoff. Local payoffs are defined almost identically to cell payoffs.</p> <p>always "AVG" means that the "AVG" expression applies to all observation dates. Four primitives, AVG, MAX, MIN, SUM, may be potentially combined to define local payoffs.</p>	<p>always "AVG"</p>
<p>global_payoff string Defines the function applied (horizontally) to local payoffs to calculate the global payoff.</p> <p>Four primitives, AVG, MAX, MIN, SUM, may be potentially combined to define global payoffs. In the opposite definition, max is an operator and MAX a payoff primitive.</p>	<p>"0.85 + max 0 (MAX - 1)"</p>
<p>payment_date date Date on which the global payoff is paid.</p>	<p>2013-11-14</p>

The logic of the above parametric representation for the sample product may be summarized as follows:

Figure 1
Logic of Possible Parametric Representation



The choice of parameter set depends on the user's requirements. The above representation, while specific to the sample contract, enables the description of several product variations, as illustrated below:

Table 3
Examples of Product Variations

Product Variation	Parameter	Parameter Value
Local payoff is difference between maximum and minimum performance, with local cap of 50%.	local_payoff	always "min 0.5 (MAX - MIN)"
Global payoff is average of local payoffs, without floor.	global_payoff	"AVG"

Note that users who are interested in analyzing only a limited number of product variations may opt for a more concise parametric definition than the one presented above. Alternatively, the definition could be generalized to cover a much broader set of products.

A Microsoft .NET / Windows Forms trade entry screen is derived from the above parametric representation, as illustrated below.

Figure 2
Sample Product Trade Entry Screen

Simulation

Simulation covers

- the execution of a contract over past, present, or future market scenarios, typically to analyze the potential performance of a product, and
- the evolution of a contract's value over a set of future dates.

Simulation of Contract's Execution (Performance Analysis)

LexiFi users may simulate the execution of a contract through time for a given market scenario, record the cash flows derived from the contract along the simulated path, and calculate summary performance metrics such as the contract's annual return.

Scenarios may be historical or forward-looking:

- historical scenarios reflect the past evolution of the contract's underlyings;
- hypothetical, forward-looking scenarios of the contract's underlyings are qualitative and may reflect the customer's view, the provider's view, or a consensus forecast.

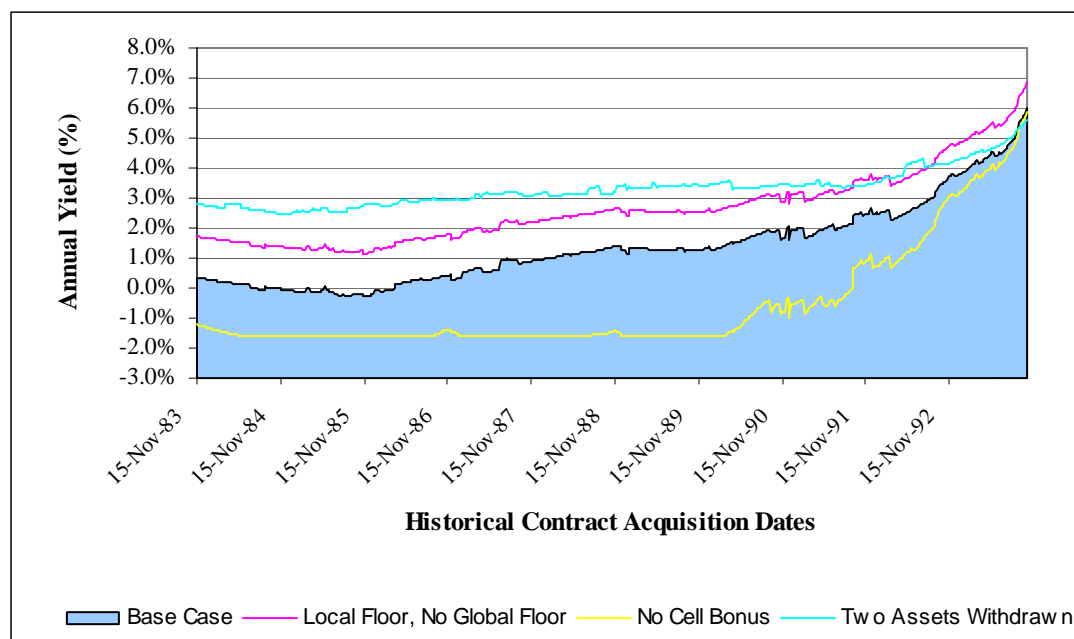
The simulated execution of a contract provides an intuitive understanding of the proposed transaction and, ultimately, helps to validate a marketing idea.

Historical Simulation

Historical simulation quantifies the performance of the proposed product as if it had been acquired in the past. The shaded area in the chart below represents the annual yield that an investor would have received if he had acquired the sample product at weekly time intervals from 1983-11-15 to 1993-11-14, the last date on which the historical acquisition of a 10-year product may be simulated, if we assume that today's date is 2003-11-14.

Figure 3

Historical Simulation of Sample Product and Three Product Variations over Ten-Year Period



The historical scenario is also run for three product variations described in the table below:

Table 4

Product Variations

Product Name (Index)	Description
Base case (0)	Sample contract.
No cell bonus (1)	Sample contract without cell bonus from the first to the seventh observation date.
Two assets withdrawn (2)	The best two (instead of one) performing assets are withdrawn from the basket on the first five observation dates. Only one asset is withdrawn from the sixth observation date.
Local floor, no global floor (3)	Local payoff changes from "AVG" to "max 0 AVG". Global payoff changes from "0.85 + max 0 MAX" to "MAX".

Users may describe scenarios explicitly with time series data or derive scenarios from one or more other scenarios. For example, users may use substitution rules when historical data is unavailable either because an asset did not exist on a simulated acquisition date or market data is missing.

Forward-Looking Simulation

The execution of a contract may also be simulated for a single acquisition date (today) and for a set of forward-looking scenarios.

We augment the set of forward-looking scenarios: in addition to the bullish spot price scenario 0 presented in Table 1 above, we define a bearish scenario 1 and a (very) bullish scenario 2. The table below displays the payoff and the annual yield of contracts 0 to 3 as well as the annual yield of the best and worst performing assets in each of the three scenarios.

Figure 4
Forward-Looking Simulation Results

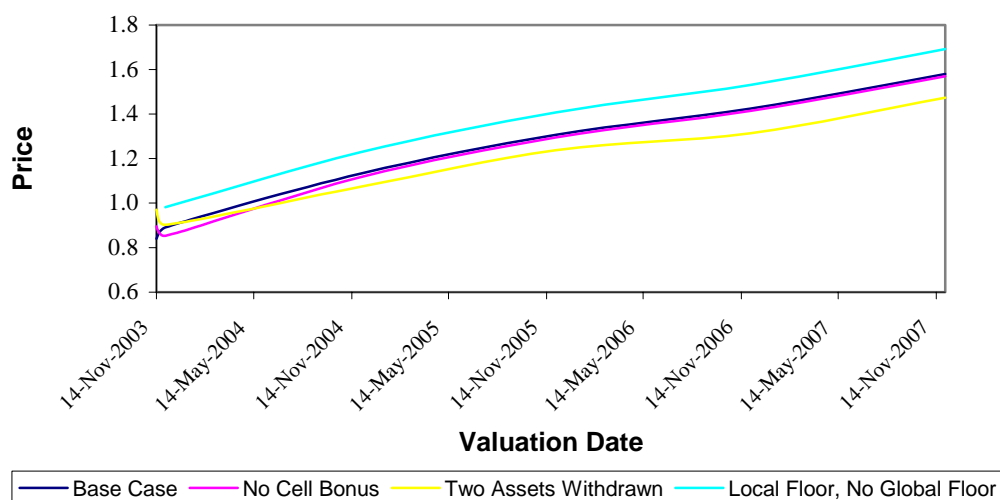
Contracts	Data	Scenarios		
		Scenario 0	Scenario 1	Scenario 2
Base Case	Payoff	186.35%	85.00%	263.55%
	Annual Yield	6.43%	-1.61%	10.19%
	Best Performer	12.03%	7.71%	14.24%
	Worst Performer	-5.47%	-9.98%	4.75%
No Cell Bonus	Payoff	184.90%	85.00%	263.55%
	Annual Yield	6.35%	-1.61%	10.19%
	Best Performer	12.03%	7.71%	14.24%
	Worst Performer	-5.47%	-9.98%	4.75%
Two Assets Withdrawn	Payoff	184.55%	123.55%	256.60%
	Annual Yield	6.33%	2.14%	9.90%
	Best Performer	12.03%	7.71%	14.24%
	Worst Performer	-5.47%	-9.98%	4.75%
Local Floor No Global Floor	Payoff	201.35%	100.00%	278.55%
	Annual Yield	7.26%	0.00%	10.80%
	Best Performer	12.03%	7.71%	14.24%
	Worst Performer	-5.47%	-9.98%	4.75%

Simulation of Future Price

The goal here is to measure a contract's value over a set of future dates. The system simulates the evolution of the contract as life cycle events unfold, records cash flows that occur between today and each future valuation date, and calculates the value of the residual contract.

The figure below illustrates the price of contracts 0 to 3, calculated on five future dates (2003-12-01, 2004-12-01, 2005-12-01, 2006-12-01, 2007-12-01), using base scenario 0 and holding other model inputs (yield curve, dividends, volatilities, correlations, etc.) constant.

Figure 5
Simulation of Future Price



The simulation of future prices helps banks to properly represent derivatives risk and enables structured product users to monitor limits and management constraints through time. It also provides a rigorous framework for calculating value at risk and potential credit exposures on a portfolio of exotic products.

Simulation Audit

LexiFi provides extensive details on how simulated fixing values were obtained. For example, LexiFi indicates that a value was available on the expected date or that a substitution rule was applied with the details of the preceding date, the following date, or both for interpolated values. When a fixing value is the result of a formula—e.g., for scenarios derived from other scenarios—all inputs of the scenario composition formula are provided.

Figure 6
Simulation Audit

```
((2004-11-15, "asset01"), ((Obs_float(106.)), Fix_from_data));
((2004-11-15, "asset02"), ((Obs_float(95.)), Fix_from_data));
((2004-11-15, "asset03"), ((Obs_float(156.)), Fix_from_data));
((2004-11-15, "asset04"), ((Obs_float(159.)), Fix_from_data));
((2004-11-15, "asset05"), ((Obs_float(98.)), Fix_from_data));
((2004-11-15, "asset06"), ((Obs_float(110.)), Fix_from_data));
((2004-11-15, "asset07"), ((Obs_float(114.)), Fix_from_data));
((2004-11-15, "asset08"), ((Obs_float(173.)), Fix_from_data));
((2004-11-15, "asset09"), ((Obs_float(105.)), Fix_from_data));
((2004-11-15, "asset10"), ((Obs_float(159.)), Fix_from_data));
((2004-11-15, "asset11"), ((Obs_float(150.)), Fix_from_data));
((2004-11-15, "asset12"), ((Obs_float(121.)), Fix_from_data));
((2004-11-15, "asset13"), ((Obs_float(99.)), Fix_from_data));
((2004-11-15, "asset14"), ((Obs_float(124.)), Fix_from_data));
((2004-11-15, "asset15"), ((Obs_float(158.)), Fix_from_data));
((2004-11-15, "asset16"), ((Obs_float(157.)), Fix_from_data));
((2004-11-15, "asset17"), ((Obs_float(109.)), Fix_from_data));
((2004-11-15, "asset18"), ((Obs_float(113.)), Fix_from_data));
((2004-11-15, "asset19"), ((Obs_float(154.)), Fix_from_data));
((2004-11-15, "asset20"), ((Obs_float(134.)), Fix_from_data));
```

The above figure displays fixing values for scenario 0 of the forward-looking simulation, on date 2004-11-15 and for assets 1 to 19. `Fix_from_data` means that data was available on that date.

Pricing

Price, Greeks, Payoff Distribution

LexiFi generates specialized pricing code for each contract with the correct sequence of numerical calculation steps. The pricing code is then linked with an in-house or LexiFi-provided model implementation.

LexiFi's Monte-Carlo framework automatically generates multi-dimensional pricing routines. The user-definable output includes price, Greeks, cash flow distribution, pricing error estimates, and runtimes.

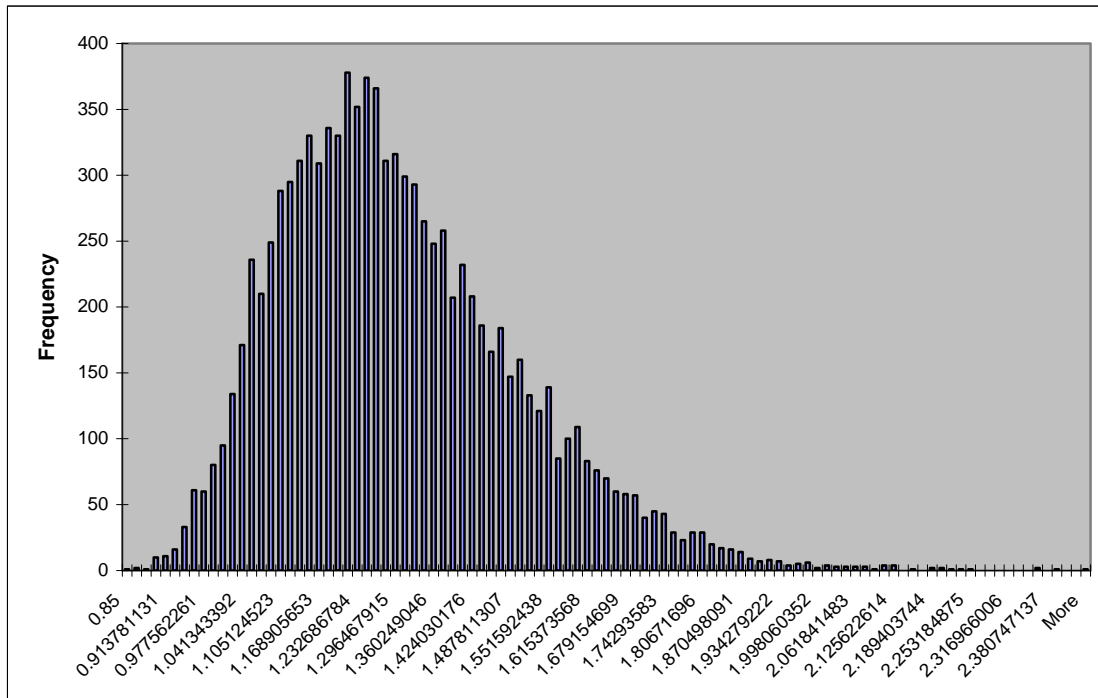
Table 4
Price and Vega

Price	0.878818142
Vega	0.271%

In the table above, vega (i.e., the sensitivity of the sample product's price to the volatility of one asset) is calculated with respect to asset 11. All values are calculated immediately after the initial fixing.

The payoff distribution is illustrated below.

Figure 7
Payoff Distribution for Monte-Carlo Model (10,000 paths)

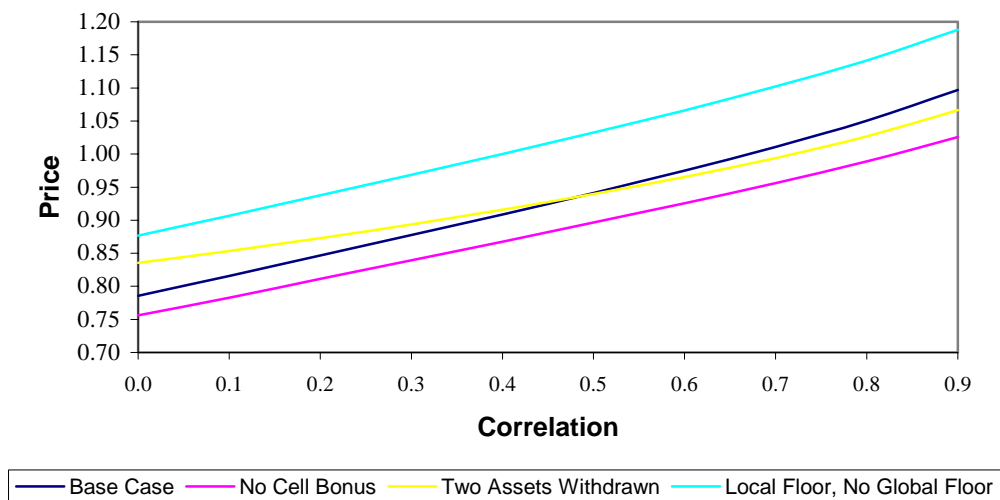


Sensitivity Analysis

Correlation

We compare the different product variations in terms of their sensitivity to an overall level of correlation between the assets. We assume that there is an overall level of correlation ρ such that the correlation between (log-) asset i and (log-) asset j equals ρ for all $i \neq j$. We then vary the level ρ between zero and one to determine the correlation sensitivity of the different products.

Figure 8
Price Sensitivity to General Correlation Level

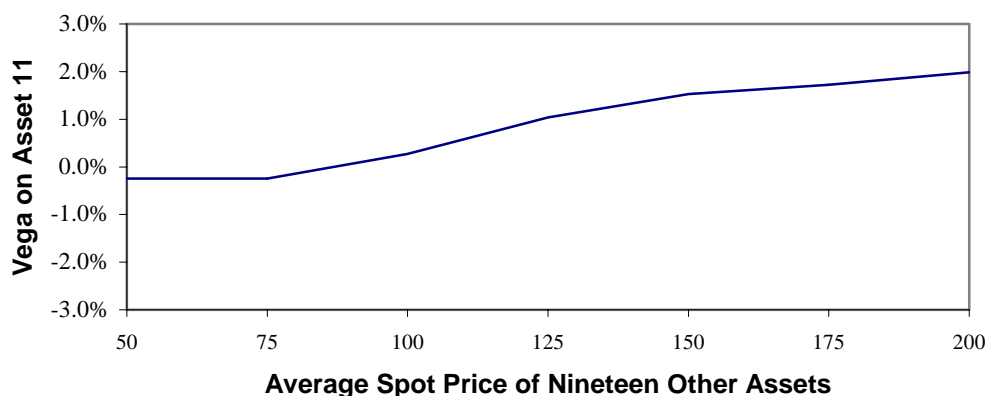


Vega

For correlation products, the vega on one asset depends on the spot level of other assets. The vega on asset 11 in the sample contract changes, depending on the average spot price of the nineteen other assets. This behavior contrasts with vanilla options where there is no correlation effect and the vega is unique.

This means that a multi-underlying option may not be hedged with a stable combination of vanilla options. Hedging must be adjusted throughout the life of the product.

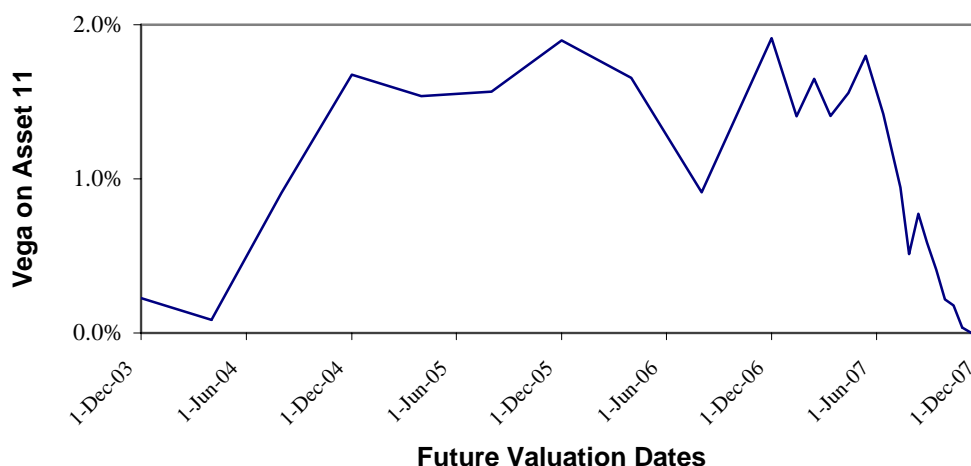
Figure 9
Vega Stability Analysis



In the above example, the price of asset 11 is fixed at 100 and the price of all other assets in the basket is moved as illustrated on the horizontal axis.

The simulation feature helps to visualize the evolution of vega along a given forward-looking scenario. The figure below shows the future value of vega on asset 11 for scenario 0.

Figure 10
Evolution of Vega on Asset 11 for Scenario 0



Vega on asset 11 goes to zero from 2007-11-14 when asset 11 is removed from the basket.

Event planning

Event Detection

LexiFi provides tools to systematically detect contract life cycle events and to analyze the potential behavior of a contract in the future. In particular, users may visualize a contract's entire execution calendar.

For example, users may wish to view the next fixing date (2004-11-15) and the list of assets that need to be fixed on that date, as illustrated below:

Figure 11
Event Planning

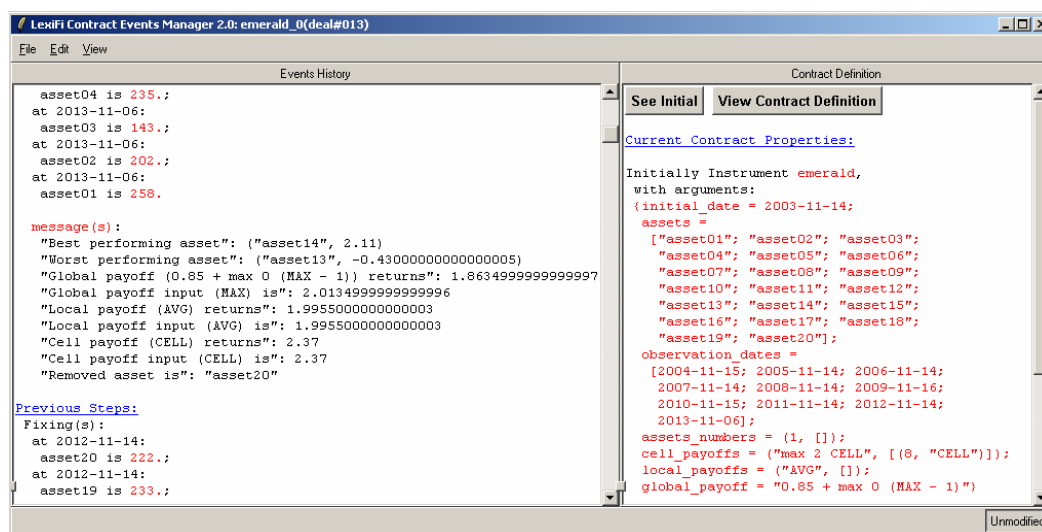
Date	Contract Type	Trade Id	Event Type	Event Description
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset01
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset02
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset03
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset04
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset05
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset06
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset07
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset08
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset09
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset10
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset11
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset12
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset13
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset14
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset15
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset16
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset17
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset18

2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset19
2004-11-15 12:00:00	Equity Derivative	base_case	Fixing	asset20

Contract Explorer

In addition, a contract explorer enables users to exhaustively analyze all potential contract execution paths. In the figure below, the contract explorer provides the inputs and outputs of the cell, local, and global payoff formulas on the last observation date, assuming base scenario 0. The contract's performance may be compared with that of the best and worst performing assets.

Figure 12
Simulation of Contract's Execution with Contract Explorer



From Prototyping to Production

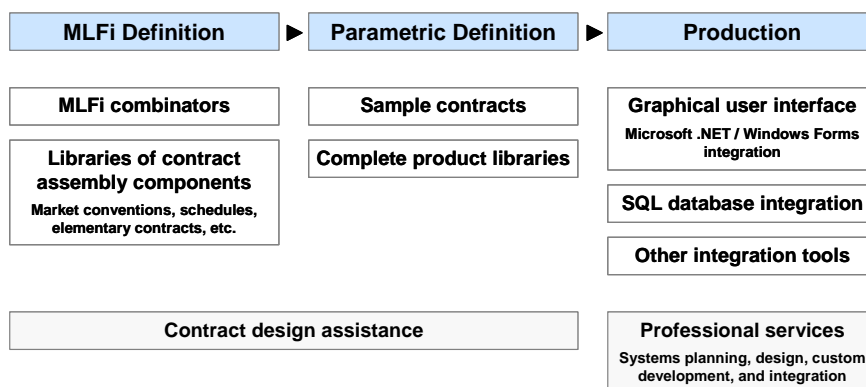
LexiFi facilitates the transition from a prototyping to a production environment.

Product Design, User Interface, and Persistence

As discussed in the Product Definition section above, the specification of entirely new products is initially written by structurers using the MlFi language. The definition is then expanded and transformed into a parametric definition that non-technical users can use. Finally, a Microsoft .NET / Windows Forms trade entry screen is derived from the parametric definition and trades are persisted in a SQL database.

The figure below summarizes the steps, tools and services provided by LexiFi for rapidly defining new products and automating their management.

Figure 13
Steps, tools, and services for commodifying new products



Rapid Prototyping and Staged Deployment of New Valuation Algorithms

MLFi may serve as a shared contract description language that generates pricing code to drive internally developed or commercially available pricing libraries. MLFi can drive both prototyping and production valuation libraries. For example, in order to rapidly respond to a customer inquiry, quantitative analysts may initially want to generate prototype code in the language of their choice. After the transaction is consummated, MLFi may generate production code in C in order to integrate the new product in the trading system. The MLFi compiler's ability to output different flavors of valuation code from the same product definition eases the programming and maintenance workload of financial engineers, and allows them to spend more time on value-added tasks such as deepening their understanding of exotic derivatives.

Reporting

LexiFi provides immediate operational reporting capabilities for all contracts, regardless of their complexity. Pre-packaged management reports cover transaction activity, event monitoring, and payments.

Users may also create precise, product-specific reports on exotic contracts with SQL-compatible tools.

Figure 14
Transaction Activity Report

Id	Trade Id	Initial Date	Assets	Observation Dates	Number Withdrawn	Cell Payoffs	Local Payoffs	Global Payoff
7	base_case	2003-11-14	asset01; asset02; asset03; asset04; asset05; asset06; asset07; asset08; asset09; asset10; asset11; asset12; asset13; asset14; asset15; asset16; asset17; asset18; asset19; asset20	2004-11-15; 2005-11-14; 2006-11-14; 2007-11-14; 2008-11-14; 2009-11-16; 2010-11-15; 2011-11-14; 2012-11-14; 2013-11-06	1	max 2 CELL, [8, CELL]	AVG	0.85 + max 0 (MAX - 1)

The above transaction activity report is tailored to the sample contract.

Conclusion

This document illustrates how structurers and salespeople may use LexiFi to develop tailored customer solutions. In particular, we demonstrate the versatility of MLFi contract definitions: a single product specification may be manipulated in a number of ways to cover diverse needs such as simulation, pricing, event planning, and operational management. LexiFi also provides features to rapidly transition from a prototyping to a production environment

LexiFi's sales support capabilities create value for financial institutions who wish to expand their structured products business. With LexiFi, you can:

- *Increase revenue opportunities.* LexiFi promotes an active dialog between the sales team and customers. Regular, quality interactions create customer intimacy and provide opportunities to develop effective customer solutions and to expand the range of structured product applications. The results are increased customer satisfaction and stronger relationships.
- *Improve skills.* LexiFi helps users to develop an intuitive understanding of structured products and their application to complex customer problems. LexiFi increases the skills, situational fluency, and comfort levels of the sales force.
- *Increase productivity.* The sales team focuses on value-added tasks and does not waste time developing complex spreadsheets. Better analytical tools also translate into improved success rates.
- *Reduce operational risk.* A formal contract definition and shared simulation results improve communication and increase transparency among stakeholders, which means fewer errors, delays, and disputes.
- *Limit legal exposure on derivatives sales.* LexiFi helps sales professionals to provide a balanced picture of the risks and potential financial consequences for the customer of accepting the terms of the proposed transaction. Fact-based sales practices reduce the risk of misrepresentation, which constitutes a potentially damaging threat for the seller.