**Lecture 2 (Forwards Markets) Assignment**

Due start of class, Wednesday September 18, 2019

**Question 1 (4 marks)**

Why are correlations of daily returns of spot vs daily returns of forward prices so high in the FX markets? What are the two requirements a market must support to enforce a high correlation across the forward curve?

The correlations are high because the FX markets admit the spot vs forward arbitrage that forces the market forward price to equal the “fair” forward price, which is a function of spot and interest rates. So when the spot price moves, all the forwards move proportionately, and correlations across the curve are high.

The two requirements are the ability to store the asset and the ability to borrow & short the asset.

**Question 2 (2 marks)**

Why is risk management more complex for an FX forwards risk manager than for an FX spot risk manager?

Because the forwards market has an extra dimension of risk: the tenor of the forward contracts. That means the portfolio is rarely exactly flat; it usually has some spread positions on, since it’s unlikely for a client to want to buy exactly the same tenor forward that another client wanted to sell.

**Question 3 (2 marks)**

Explain why risk to FX forward points can be expressed as risk to non-USD interest rates.

Forward(T) = S \* exp((R(T)-Q(T))\*T), and forward point = Forward(T)-S, which is approximately equal to S \* ( R(T) – Q(T) ) \* T. So there are three dimensions of risk in a forward point: spot S, and the two interest rates (of which one is the USD rate).

Other risk reports already report risk to spot and the USD interest rates. So the one dimension of risk that’s special to FX forward points is the risk to the non-USD interest rate curve.

**Question 4 (4 marks)**

Assume a portfolio has just one FX forward position in it, settling on a date T which lies between two benchmark settlement dates T1 and T2. Derive the notionals N1 and N2 of the benchmark forwards which hedge the portfolio risk assuming triangle shocks to the benchmark non-USD interest rates, as shown on page 24 of the lecture notes.

We start by defining the price of three different forwards: one settling at T, one settling at T1, and the last settling at T2:

We’ll construct a portfolio that’s long one unit of the forward to T, and short N1 and N2 units of the forwards to T1 and T2. Then we’ll solve for N1 and N2 such that the risk to moving rates Q1 and Q2 (assuming a triangle-shock impact to QT in each case) is zero.

Under the triangle shock for Q1, QT moves by a fractional amount (T2-T)/(T2-T1). Similarly, under the triangle shock for Q2, QT moves by a fractional amount (T-T1)/(T2-T1). Using these for the partial derivatives of QT, and the expressions for the prices of the three forwards, we get

and these simplify to the expressions from page 24 when the initial Qs are all equal to a constant Q:

**Question 5 (4 marks)**

Explain principal component analysis and factor models, focusing on the differences between the two approaches to reduce dimensionality.

Principal component analysis and factor models are two different techniques for reducing dimensionality in a high-dimensional system. In this example we were looking at these techniques as ways of reducing the dimensionality of non-USD interest rate curve hedging for a book of FX forwards.

Principal component analysis is a non-parametric method for discovering the main factors driving the moves in the interest rate curve. It involves calculating the historical covariances of day-on-day changes in interest rates of different tenors (assuming that each day’s sample for a given tenor is drawn from the same distribution on each day of the historical period), then orthogonalizing that covariance matrix. The eigenvectors represent the orthogonal shocks to the curve, and the eigenvalues represent the relative sizes of those shocks. Principal component analysis involves picking off the top N largest eigenvalues and approximating all moves in the curves as being linear combinations of the respective eigenvectors. The shapes of the shocks are an output of the calculation and are non-parametric.

A factor model assumes a stochastic differential equation driving interest rates of different tenors, where all tenors are shocked by a small number of Brownian motion factors. The shocks in a factor model are parametric: their functionality forms are determined by the model parameter (for example, exponential shocks).

The main difference between the two approaches is that PCA results in non-parametric shocks, whereas a factor model has parametric shocks. Non-parametric shocks give more flexibility in terms of shape of the shocks, but also less robustness (historical data artifacts can lead to odd shapes). Parametric shocks are more robust and easier for traders to understand, but have less flexibility that PCA’s non-parametric shocks.

**Question 6 (10 marks)**

This programming question will try to determine whether using a factor-based approach to reducing dimensionality is better than an ad hoc method.

We start by assuming a toy market: spot = 1, asset currency interest rate curve = Q(T) = flat at 3%, and denominated currency interest rate curve = R(T) = flat at 0%. We assume two benchmark dates, T1 = 0.25y and T2 = 1y; we will use forwards to those settlement dates to hedge the forward rate risk (or equivalently, the risk to moves in the asset currency interest rate) of our portfolio.

In the toy market, we assume that we know the dynamics of the asset currency interest rate:

where =1%/sqrt(yr), =0.8%/sqrt(yr), =0.5/yr, =0.1/yr, and =-0.4.

The portfolio to hedge has one position: a unit asset-currency notional of a forward contract settling at time T. You’ll try this for values of T in [0.1,0.25,0.5,0.75,1,2] to see how performance changes for portfolios with risk to different tenors.

You will try three different hedging strategies: one where you choose the hedge notionals (of forwards settling at times T1 and T2) based on the triangle shock we discussed in class (though as there are only two benchmarks here, the T1 shock will be flat before T1 and the T2 shock will be flat after T2); one where the notionals are set to hedge the actual two shocks from the factors described above; and lastly, one where you don’t hedge at all.

The result should show that setting hedge notionals based on the true factor shocks should provide a better hedge performance than based on the ad hoc triangle shocks. You should analyze just how much better that performance is.

Run a Monte Carlo simulation where you do the following on each run, for each value of T, for each of the three hedging strategies described above:

1. Construct a portfolio long 1 unit of the forward settling at time T
2. Add in the hedges: two forwards, settling at times T1 and T2, with notionals set to hedge the portfolio (either against the two triangle shocks or against the two factor shocks). Don’t bother adding the hedges in the third hedge scenario where we leave the portfolio unhedged.
3. Simulate the portfolio forward a time dt=0.001y. That will result in the asset-currency rates moving according to the factor model described above, which shocks the benchmark rates for tenors T1 and T2, and for the portfolio’s risk tenor T. Determine the PNL realized.

Then construct the PNL distributions for the three hedging approaches. The unhedged version is the benchmark: you should compare how much more effectively the PNL standard deviation is reduced by hedging according to the true factors vs hedging according to the ad hoc triangle shocks.

Do this for all the values of T listed above, and discuss your results.

You should deliver this as Python scripts saved in the WST Python environment. Save them with a naming convention like baruch/<your email address with “@” replaced by “.”>/assig2.py, and push them to production from the Git Console.

Before writing any code we need to figure out the hedge notionals N1 and N2 of the two hedge forwards when the shocks we hedge are the actual shocks from the two factor model.

We modify the result from Question 4 to show that

We need to figure out the partial derivatives and when the shocks come from the factor model. The first one is how much moves when moves a small amount but is held constant; the second one is the reverse.

From the factor model we know that

So, for a given pair of rate moves and , we can figure out the Brownian motion shocks and that result in those rate moves.

First we can figure out the Brownian shocks when is non-zero but is zero:

This solves to:

We can then translate these Brownian shocks, via the factor model, into the resulting shock for to get .

Similarly we can derive the Brownian shocks under the factor model to make zero and nonzero:

and again can plug those into the factor model to find .

Once we have those two partial derivatives under the factor model, we can plug them into the expressions for the two hedge notionals.

This is implemented in the Python script wst/baruch/assig2.py. It figures out the hedge notionals required and then runs a Monte Carlo simulation to calculate the PNL standard deviation.

The results are below in a table showing PNL standard deviation\*105 (just rescaled to make the numbers closer to 1):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **T** | **No Hedge** | **Triangle Hedge** | **Factor Hedge** | **Triangle Efficiency** | **Factor Efficiency** |
| 0.10 | 0.3023 | 0.0204 | 0.0000 | 93.3% | 100.0% |
| 0.25 | 0.7140 | 0.0000 | 0.0000 | 100.0% | 100.0% |
| 0.50 | 1.3071 | 0.0179 | 0.0000 | 98.6% | 100.0% |
| 0.75 | 1.8096 | 0.0256 | 0.0000 | 98.6% | 100.0% |
| 1.00 | 2.2454 | 0.0000 | 0.0000 | 100.0% | 100.0% |
| 2.00 | 3.6166 | 1.3080 | 0.0003 | 63.8% | 100.0% |
|  |  |  |  |  |  |

The “factor” hedge is essentially perfect: after the hedge the PNL standard deviation is vanishingly small. This is as expected: there are only two drivers of moves in the curve, and if you hedge exactly as they are expected to move, you should be hedging all the risk (to first order anyways).

The triangle hedge does a very good job in between the two benchmark hedging tenors (T=0.25 and T=1.00), reducing the PNL standard deviation to less than 1.5% of its value when unhedged.

It does somewhat worse for short tenors (T=0.10), but still reasonably well; but it performs quite poorly for long tenors (T=2.00). That’s because the triangle model assumes that the curve will move in parallel from 1y and out when it calculates the notional of the 1y forward to hedge the portfolio; but the real move for the 2y tenor will be less than that of the 1y tenor because of the mean reversion, so the triangle shock results in too large a notional of the 1y forward hedge.

The conclusion: the triangle shock approach to risk calculations is accurate so long as the benchmark tenors used for the hedges span most of the portfolio risk. That’s a relief: it means we can construct risk reports using fairly simple-minded shocks and get about the same hedging performance we’d get with a much more complicated factor model.

Python code to reproduce:

"""

Copyright: Copyright (C) 2014 Washington Square Technologies - All Rights Reserved

Description: Functions for testing hedging power of different shock forms in hedging

an FX forward with benchmark FX forwards.

"""

import math

import scipy

def hedge\_stddev(sigma1, sigma2, beta1, beta2, rho, T, T1, T2, spot, Q, dt, nruns, seed, hedge\_model):

'''Runs a Monte Carlo simulation hedging a portfolio with an

FX forward settling at time T with FX forwards settling at

times T1 and T2. Here we're hedging the non-USD interest rate

sensitivity, and we'll assume that the asset currency is the

non-USD interest rate. The rate curve moves according to a

parametric 2f model

dQ(T) = sigma1 exp(-beta1 T) dz1(dt) + sigma2 exp(-beta2 T) dz2(dt)

E[dz1(t) dz2(t)] = rho dt

It returns the standard deviation of the PNL after hedging.

sigma1: the volatility of the first parameter

sigma2: the volatility of the second parameter

beta1: the mean reversion strength of the first parameter

beta2: the mean reversion strength of the second parameter

rho: the correlation between the Brownian shocks

T: the time to settlement of the unit FX forward position

T1: the time to settlement of the near hedge forward

T2: the time to settlement of the far hedge forward

spot: the FX spot rate

Q: the initial asset currency discount rate; assumed to be flat across the curve at the start

dt: the length of the time step the simulation advances over

nruns: the number of Monte Carlo runs

seed: the RNG seed

hedge\_model: the string name of the hedging model to use. "Factor" means hedge according to

the same factor model that drives moves in the curve; "Triangle" means hedge according

to the triangle shock model; and "None" means no hedging at all.

'''

# initialize the random number generator with the seed passed in. This ensures that every

# run will give the same results, so you can reproduce values.

scipy.random.seed(seed)

# generate the Brownian motion shocks. These are scipy.arrays with nruns elements; we're

# doing vectorized calculations again to make the calculation more efficient.

sqrtdt = math.sqrt(dt)

r1 = scipy.random.normal(0, sqrtdt, nruns)

r2 = scipy.random.normal(0, sqrtdt, nruns)

dz1s = r1

dz2s = rho \* r1 + math.sqrt(1 - rho \* rho) \* r2

# generate the rate shocks for the three tenors: T, T1, and T2

dQTs = sigma1 \* math.exp(-beta1 \* T) \* dz1s + sigma2 \* math.exp(-beta2 \* T) \* dz2s

dQ1s = sigma1 \* math.exp(-beta1 \* T1) \* dz1s + sigma2 \* math.exp(-beta2 \* T1) \* dz2s

dQ2s = sigma1 \* math.exp(-beta1 \* T2) \* dz1s + sigma2 \* math.exp(-beta2 \* T2) \* dz2s

# figure out the hedge notionals; this depends on the hedging model we're using

if hedge\_model == 'None':

# no hedging; hedge quantities are zero

N1 = N2 = 0

elif hedge\_model == 'Triangle':

# use triangle rate shocks

if T <= T1:

# all rates for tenors < T1 are moved in parallel when Q1 is shocked; Q(T)

# gets no bump at all when Q2 is shocked

N1 = T / T1 \* math.exp(-Q \* (T - T1))

N2 = 0

elif T >= T2:

# all rates for tenors > T2 are moved in parallel when Q2 is shocked; Q(T)

# gets no bump at all when Q1 is shocked

N1 = 0

N2 = T / T2 \* math.exp(-Q \* (T - T2))

else:

# T is between T1 and T2 and gets a triangle shock from both Q1 and Q2 shocks

N1 = (T2 - T) / (T2 - T1) \* T / T1 \* math.exp(-Q \* (T - T1))

N2 = (T - T1) / (T2 - T1) \* T / T2 \* math.exp( Q \* (T2 - T))

elif hedge\_model=='Factor':

# use shocks based on the factor model that's driving moves in the curve

# first shock: dQ1=1 unit, dQ2=0; figure out dz1 and dz2 that result in this move

dz1 = -1 / sigma1 \* math.exp(beta1 \* T2 - beta2 \* (T2 - T1)) / (1 - math.exp((beta1 - beta2) \* (T2 - T1)))

dz2 = 1 / sigma2 \* math.exp(beta2 \* T1)/(1 - math.exp((beta1 - beta2)\*(T2 - T1)))

# figure out the resulting shock to Q(T); this gets us dQ(T)/dQ1

dQTdQ1 = sigma1 \* math.exp(-beta1 \* T) \* dz1 + sigma2 \* math.exp(-beta2 \* T) \* dz2

# second shock: dQ1=0, dQ2=1 unit; similarly get dz1 and dz2 that result in this move,

# and then get dQ(T)/dQ2

dz1 = -1 / sigma1 \* math.exp(beta1 \* T1 + beta2 \* (T2 - T1)) / (1 - math.exp((beta2 - beta1) \* (T2 - T1)))

dz2 = 1 / sigma2 \* math.exp(beta2 \* T2) / (1 - math.exp((beta2 - beta1) \* (T2 - T1)))

dQTdQ2 = sigma1 \* math.exp(-beta1 \* T) \* dz1 + sigma2 \* math.exp(-beta2 \* T) \* dz2

# use these two partial derivatives to get the two hedge notionals

N1 = dQTdQ1 \*T / T1 \* math.exp(-Q \* (T - T1))

N2 = dQTdQ2 \*T / T2 \* math.exp( Q \* (T2 - T))

else:

raise ValueError('hedge\_model must be "None", "Triangle", or "Factor"; not "' + hedge\_model + '"')

# calculate the portfolio PNLs under the rate shocks; for this we use the function for

# the price of a forward settling at time t, v(t) = S exp(-Q(t) t) - K exp(-R(t) t). The

# change in the price of that forward under a rate shock dQ is then

# dv(t) = S ( exp(-(Q+dQ) t) - exp(-Q t) ) - ie the PNL doesn't depend on the forward strike

# K or the denominated (USD) discount rate R.

pnls = spot \* (scipy.exp(-(Q + dQTs) \* T) - scipy.exp(-Q \* T))

pnls -= N1 \* spot \* (scipy.exp(-(Q + dQ1s) \* T1) - scipy.exp(-Q \* T1))

pnls -= N2 \* spot \* (scipy.exp(-(Q + dQ2s) \* T2) - scipy.exp(-Q \* T2))

# return the simulation PNLs

return pnls.std()

def test():

'''Test out the MC sim'''

# set up the model parameters per the assignment question

sigma1 = 0.01

sigma2 = 0.008

beta1 = 0.5

beta2 = 0.1

rho = -0.4

T1 = 0.25

T2 = 1

spot = 1

Q = 0.03

dt = 1e-3

nruns = 100000

seed = 1

# run through a range of values of T and look at the results for the

# three different hedging strategies

Ts = [0.1, 0.25, 0.5, 0.75, 1, 2]

hedge\_models = ['None','Triangle','Factor']

for T in Ts:

print('Forward time',T)

for hedge\_model in hedge\_models:

std = hedge\_stddev(sigma1, sigma2, beta1, beta2, rho, T, T1, T2, spot, Q, dt, nruns, seed, hedge\_model)

print(' hedge model', hedge\_model, 'standard deviation is', std \* 1e4)

print()

if \_\_name\_\_=="\_\_main\_\_":

test()