Examples with algorithms for the course “Financial Computing with C++”

Master of Science in Computational Finance
Carnegie Mellon University

Spring 2025

Contents

Data curves for financial models

While implementing the functions below, you need to account for the singularities of the type 0/0.

Yield curve computed from discount curve

Input:

$D = (D(t))_{t \geq t_0}$: the discount curve.

t_0 : the initial time.

Output: the continuously compounded yield curve $\gamma = (\gamma(t))_{t \geq t_0}$.

We recall that

$$D(t) = e^{-\gamma(t)(t-t_0)}, \quad t \geq t_0.$$

Algorithm. To handle the singularity at $t \approx t_0$, $t \geq t_0$, we take sufficiently small ϵ (say, $\epsilon = 10^{-10}$) and denote

$$\tau(t) \triangleq \max(t - t_0, \epsilon).$$

We then compute the yield as

$$\gamma(t) = -\frac{1}{\tau(t)} \log D(t_0 + \tau(t)) = -\frac{1}{\tau(t)} \operatorname{logp1}\left(D(t_0 + \tau(t)) - 1\right),$$

where the function $\operatorname{logp1}(x) \triangleq \log(1 + x)$ is a part of STL and improves the standard log function when $x \approx 0$.

The Nelson-Siegel discount curve

Input:

$(c_i)_{i=0,1,2}$: the coefficients of the Nelson-Siegel yield curve.

$\lambda > 0$: the mean-reversion rate.

t_0 : the initial time.

Output: the discount curve

$$D(t) = e^{-\gamma(t)(t-t_0)}, \quad t \geq t_0,$$

where the yield curve has the Nelson-Siegel form:

$$\gamma(t) = c_0 + c_1 \frac{1-e^{-\lambda(t-t_0)}}{\lambda(t-t_0)} + c_2 \left(\frac{1-e^{-\lambda(t-t_0)}}{\lambda(t-t_0)} - e^{-\lambda(t-t_0)} \right), \quad t \geq t_0.$$

Algorithm. We write the yield curve as

$$\gamma(t) = c_0 + c_1 g_1(\alpha(t)) + c_2 g_2(\alpha(t)), \quad t \geq t_0,$$

where, for $t \geq t_0$ and $x \in \mathbb{R}$,

$$\begin{aligned} \alpha(t) &= \lambda(t-t_0), \\ g_1(x) &= \frac{1}{x}\left(1-e^{-x}\right), \\ g_2(x) &= g_1(x) - e^{-x}. \end{aligned}$$

The functions g_1 and g_2 have a removable singularity at $x = 0$. They are implemented in `cfl` as `cfl::GetShape1` and `cfl::GetShape2`.

Discount curve for the Vasicek model of interest rates

Input:

$\theta :$ the constant drift term (θ/λ is the mean-reversion level).

$\lambda \geq 0 :$ the mean-reversion rate.

$\sigma > 0 :$ the volatility.

$r_0 = r(t_0) :$ the initial short-term interest rate.

$t_0 :$ the initial time given as year fraction.

Output:

$D = (D(t))_{t \geq t_0} :$ the discount curve in the Vasicek model of interest rates.

In the Vasicek model, under the risk-neutral probability, the short-term interest rate evolves as

$$dr(t) = (\theta - \lambda r(t))dt + \sigma dW_t, \quad t \geq t_0,$$

where W is a Brownian motion. The discount curve is given by

$$D(t) = \mathbb{E}\left(e^{-\int_{t_0}^{t} r(s)ds}\right) = e^{-\gamma(t)(t-t_0)}, \quad t \geq t_0.$$

One can show that the yield curve $\gamma = \gamma(t)$ has the form

$$\gamma(t) = r_0 A(t) + \frac{\theta}{\lambda}(1 - A(t)) - \frac{\sigma^2}{2\lambda^2}(1 - 2A(t) + B(t)), \quad t \geq t_0,$$

where

$$A(t) = \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t-t_0)}, \quad B(t) = \frac{1 - e^{-2\lambda(t-t_0)}}{2\lambda(t-t_0)}.$$

Algorithm. We write the yield curve as

$$\gamma(t) = r_0 g_1(\alpha(t)) + \theta(t-t_0)g_3(\alpha(t)) - \frac{1}{2}\sigma^2(t-t_0)^2 g_4(\alpha(t)), \quad t \geq t_0,$$

where, for $t \geq t_0$ and $x \in \mathbb{R}$,

$$\begin{aligned}
\alpha(t) &= \lambda(t-t_0), \\
g_1(x) &= \frac{1}{x}\left(1 - e^{-x}\right), \\
g_3(x) &= \frac{1}{x}\left(1 - g_1(x)\right), \\
g_4(x) &= \frac{1}{x^2}\left(1 - 2g_1(x) + g_1(2x)\right).
\end{aligned}$$

The functions g_1, g_3, and g_4 have a removable singularity at $x = 0$. They are implemented in `cfl` as `cfl::GetShape1`, `cfl::GetShape3`, and `cfl::GetShape4`.

Forward price curve for a coupon bond

Input:

q : the coupon rate.

δs : the time interval between coupon payments.

T : the maturity.

$B = (B(t_0, t))_{t \geq t_0}$: the discount curve.

t_0 : the initial time.

is_clean : the boolean parameter specifying the type of the prices: "clean" or "dirty". The dirty price is the actual amount paid in a transaction. The clean price is the difference between the dirty price and the accrued interest. If t is the settlement time and s_m is the previous coupon or issue time, then the accrued interest is computed as

$$A(t) = q(t - s_m), \quad s_m \leq t < s_{m+1} = s_m + \delta s.$$

Output:

$F = (F(t_0, t))_{t \in [t_0, T]}$: the curve of forward prices for the bond computed at t_0 for maturities $t \in [t_0, T]$.

The bond pays coupons $q\delta s$ at times $(s_m)_{m=1,\dots,M}$ such that

$$s_0 \leq t_0 < s_1 \leq t_0 + \delta s, \quad s_{m+1} - s_m = \delta s, \quad s_M = T.$$

The bond also pays notional $N = 1$ at maturity T. The buyer of the forward contract pays "dirty" forward price $F(t_0, t)$ at delivery time t and receives the bond, that is, gets the notional $N = 1$ at maturity T and coupons $q\delta s$ at times $s_m > t$.

Algorithm. We use the following notations:

$F(t_0, t)$: the dirty forward bond price computed at t_0 for delivery at t.

$P(t)$: the dirty spot bond price at t (= the value of coupons $q\delta s$ at times $s_m > t$ and notional 1 at T).

$B(s, t)$: the discount factor computed at s for maturity t.

At delivery time t, the long position in the forward contract has value

$$V(t) = P(t) - F(t_0, t) = q\delta s \sum_{s_m > t} B(t, s_m) + B(t, T) - F(t_0, t).$$

At issue time t_0, the long position has value 0. Using replication, we deduce that

$$0 = V(t_0) = q\delta s \sum_{s_m > t} B(t_0, s_m) + B(t_0, T) - B(t_0, t)F(t_0, t).$$

It follows that

$$F(t_0, t) = \frac{1}{B(t_0, t)} \left(q\delta s \sum_{s_m > t} B(t_0, s_m) + B(t_0, T) \right).$$

For the clean price, we have that

$$F^{\text{clean}}(t_0, t) = F(t_0, t) - qa(t),$$

where $a(t)$ is the time elapsed from the previous coupon (or issue) time:

$$a(t) = t - \max\{s_m : \ s_m \leq t, \ m = 0, \ldots, M\}.$$

One-dimensional root finding

Price of a call in the Black model

Input:

$K > 0:$ the strike.

$T > 0:$ the time to maturity.

$D > 0:$ the discount factor.

$F > 0:$ the forward price.

$\sigma > 0:$ the volatility.

Output: the price C of the call option in the Black model.

We recall that

$$\begin{aligned} C &= D\mathbb{E}\left(\max(F\exp(\xi\sigma\sqrt{T} - \frac{1}{2}\sigma^2 T) - K, 0)\right) \\ &= D(F\Phi(d_1) - K\Phi(d_2)), \end{aligned}$$

where $\xi \sim \mathcal{N}(0,1)$ is the standard Gaussian random variable with mean 0 and variance 1,

$$\Phi(x) = \mathbb{P}\left(\xi \leq x\right), \quad x \in \mathbb{R},$$

is the standard Gaussian cumulative distribution function (cdf), and

$$\begin{aligned} d_1 &= \frac{\ln(F/K)}{\sigma\sqrt{T}} + \frac{1}{2}\sigma\sqrt{T} \\ d_2 &= \frac{\ln(F/K)}{\sigma\sqrt{T}} - \frac{1}{2}\sigma\sqrt{T} = d_1 - \sigma\sqrt{T}. \end{aligned}$$

Algorithm. To implement Gaussian cdf $\Phi = \Phi(x)$ we use the function

`gsl_cdf_ugaussian_P`

from GSL, declared in `<gsl_cdf.h>`.

Vega for a call option in the Black model

Input:

$K > 0:$ the strike.

$T > 0:$ the time to maturity.

$D > 0:$ the discount factor.

$F > 0:$ the forward price.

$\sigma > 0:$ the volatility.

Output: the vega $\nu = \frac{\partial}{\partial\sigma}C$ of the call option in the Black model. Here, C is the price of the call option.

Algorithm. We deduce that

$$\begin{aligned}
\nu &= \frac{\partial}{\partial\sigma} D\mathbb{E}\left(\max(F\exp(\xi\sigma\sqrt{T} - \frac{1}{2}\sigma^2 T) - K, 0)\right) \\
&= D\frac{\partial}{\partial\sigma}(F\Phi(d_1) - K\Phi(d_2)) \\
&= \frac{D}{\sigma}(K\phi(d_2)d_1 - F\phi(d_1)d_2) \\
&= DF\sqrt{T}\phi(d_1) = DK\sqrt{T}\phi(d_2),
\end{aligned}$$

where $\xi \sim \mathcal{N}(0,1)$ is the standard Gaussian random variable with mean 0 and variance 1,

$$\begin{aligned}
\phi(x) &= \frac{1}{\sqrt{2\pi}}\exp(-\frac{x^2}{2}), \quad x \in \mathbb{R}, \\
\Phi(x) &= \int_{-\infty}^{x} \phi(y)dy = \mathbb{P}\left(\xi \leq x\right), \quad x \in \mathbb{R},
\end{aligned}$$

are the standard Gaussian density and distribution functions,

$$\begin{aligned}
d_1 &= \frac{\ln(F/K)}{\sigma\sqrt{T}} + \frac{1}{2}\sigma\sqrt{T}, \\
d_2 &= \frac{\ln(F/K)}{\sigma\sqrt{T}} - \frac{1}{2}\sigma\sqrt{T} = d_1 - \sigma\sqrt{T},
\end{aligned}$$

and we used the identities:

$$\begin{aligned}
\frac{\partial}{\partial\sigma}d_1 &= -\frac{\ln(F/K)}{\sigma^2\sqrt{T}} + \frac{1}{2}\sqrt{T} = -\frac{d_2}{\sigma}, \\
\frac{\partial}{\partial\sigma}d_2 &= -\frac{d_1}{\sigma}, \\
\frac{\phi(d_1)}{\phi(d_2)} &= \exp(-\frac{1}{2}(d_1^2 - d_2^2)) = \exp(-\frac{1}{2}(d_1 - d_2)(d_1 + d_2)) \\
&= \exp(-\ln(F/K)) = \frac{K}{F}.
\end{aligned}$$

To implement the Gaussian pdf $\phi = \phi(x)$ we use the function

`gsl_ran_gaussian_pdf`

from GSL, declared in `<gsl_randist.h>`.

Implied volatility for a call option in the Black model with Newton method

Input:

$K > 0:$ the strike.

$T > 0:$ the time to maturity.

$D > 0:$ the discount factor.

$F > 0:$ the forward price.

$C > 0:$ the price of the call,

$\sigma_0 > 0:$ the initial guess for the volatility.

$\epsilon > 0:$ the absolute error for the price.

Output: the implied volatility σ such that

$$|C - f(\sigma)| < \epsilon,$$

where $f(\sigma)$ is the value of the call option in the Black model with volatility σ:

$$f(\sigma) = D\mathbb{E}\left(\max(F\exp(\xi\sigma\sqrt{T} - \frac{1}{2}\sigma^2 T) - K, 0)\right).$$

Here $\xi \sim \mathcal{N}(0,1)$ is the standard Gaussian random variable with mean 0 and variance 1.

We find the implied volatility σ using the Newton (gradient descent) method.

Algorithm. To find root σ we use the Newton method:

$$\sigma_{n+1} = \sigma_n - \frac{f(\sigma_n) - C}{f'(\sigma_n)}, \quad n \geq 0.$$

We stop as soon as $|f(\sigma_{n+1}) - C| < \epsilon$.

Remark. The idea behind the Newton (the gradient descent) method comes from the first order expansion of function

$$g(\sigma) = f(\sigma) - C,$$

in a neighborhood of a root σ^* where $g(\sigma^*) = 0$:

$$g(\sigma) = f(\sigma) - C \approx g(\sigma^*) + g'(\sigma)(\sigma - \sigma^*) = f'(\sigma)(\sigma - \sigma^*), \quad \sigma \approx \sigma^*,$$

so that

$$\sigma^* \approx \sigma - \frac{g(\sigma)}{g'(\sigma)} = \sigma - \frac{f(\sigma) - C}{f'(\sigma)}.$$

Implied volatility for a call option in the Black model with `cfl::RootD`

Input:

$K > 0:$ the strike.

$T > 0:$ the time to maturity.

$D > 0:$ the discount factor.

$F > 0:$ the forward price.

$C > 0:$ the price of the call.

$\sigma_0 > 0:$ the initial guess for the volatility.

`cfl::RootD`: the root-finding routing of polishing type. It requires the computation of the function and its derivative.

Output:

σ : the implied volatility of the call option with the given parameters computed in the Black model:

$$C = D\mathbb{E}\left(\max(F\exp(\xi\sigma\sqrt{T} - \frac{1}{2}\sigma^2 T) - K, 0)\right).$$

Here $\xi \sim \mathcal{N}(0,1)$ is the standard Gaussian random variable with mean 0 and variance 1.

Algorithm. We run the root-finding algorithm with the function

$$\begin{aligned} f(\sigma) &= D\mathbb{E}\left(\max(F\exp(\xi\sigma\sqrt{T} - \frac{1}{2}\sigma^2 T) - K, 0)\right) - C \\ &= \text{theoretical price}(\sigma) - \text{market price}, \end{aligned}$$

its derivative $f'(\sigma)$ (the vega), and the initial guess σ_0.

Implied volatility for a portfolio of call options in the Black model with `cfl::Root`

Input:

Portfolio of call options: $(N_m, K_m, T_m)_{m=1,\dots,M}$, where

N_m : the number of options (can be negative, but better to be positive to avoid counter-intuitive results).

K_m : the strike.

T_m : the maturity according to the absolute calendar.

C : the market price for the portfolio.

$(D(t))_{t\geq t_0}$: the discount curve.

$(F(t))_{t\geq t_0}$: the forward curve.

t_0 : the initial time.

σ_0 : the initial lower bound for the volatility.

σ_1 : the initial upper bound for the volatility, $\sigma_1 > \sigma_0$.

`cfl::Root:` the root-finding routing of bracketing type. It relies on the computation of the function only.

Output:

σ : the implied volatility of the portfolio of call options in the Black model:

$$\text{theoretical price}\,(\sigma) = \text{market price}.$$

Algorithm. We run the root-finding algorithm of bracketing type with the initial interval $[\sigma_0, \sigma_1]$ for the function

$$f(\sigma) = \sum_{m=1}^{M} N_m g(K_m, D(T_m), F(T_m), T_m - t_0, \sigma) - C,$$

where $g(K, D, F, T, \sigma)$ is the Black and Scholes price for the call option with strike K and time to maturity T under discount factor D, forward price F, and volatility σ.

Interpolation of data curves

While implementing the functions below, you need to account for the singularities of the type 0/0.

Discount curve obtained by log interpolation

Input:

$(t_m)_{m=1,\dots,M}:$ the maturities, $t_m < t_{m+1}$,

$(D_m)_{m=1,\dots,M}:$ the discount factors,

$t_0:$ the initial time, $t_0 < t_1$,

$\mathcal{I}:$ an interpolation method, a class `cfl::Interpolation`.

Output: the discount curve

$$D(t) = \exp(L(t)), \quad t \in [t_0, t_M],$$

where the function $L = L(t)$ is the $\mathcal{I}$-interpolation of the logs of the market discount factors:

$$L(t) = \mathcal{I}((t_m)_{m=0,1,\dots,M}, (\log D_m)_{m=0,\dots,M}), \quad D_0 = 1.$$

Algorithm. We divide the algorithm into steps.

Step 1 (The logarithms of discount factors). We compute the logarithms of market discount factors:

$$y_m = \log D_m, \quad m = 0, \dots, M,$$

where $D_0 = D(t_0) = 1$ and thus, $y_0 = \log 1 = 0$.

Step 2 (Interpolation). We compute the interpolated function

$$L(t) = \mathcal{I}((t_m)_{m=0,\dots,M}, (y_m)_{m=0,\dots,M}), \quad t \in [t_0, t_M].$$

Step 3. We return

$$D(t) = \exp(L(t)), \quad t \in [t_0, t_M].$$

Forward curve obtained by interpolation of cost-of-carry rates

Input:

> S_0 : the spot price.
>
> $(t_m)_{m=1,\dots,M}$: the maturities, $t_m < t_{m+1}$.
>
> $(F_m)_{m=1,\dots,M}$: the forward prices.
>
> $q_0 = q(t_0)$: the initial cost-of-carry rate.
>
> t_0 : the initial time = the issue time for forwards, $t_0 < t_1$.
>
> $\mathcal{I}$: an interpolation method, a class `cfl::Interpolation`.

Output: the forward curve

$$F(t) = S_0 \exp(q(t)(t - t_0)), \quad t \in [t_0, t_M],$$

where the cost-of-carry function $q = q(t)$ is the $\mathcal{I}$-interpolation of the market cost-of-carry rates:

$$\begin{aligned} q(t) &= \mathcal{I}((t_m)_{m=0,1,\dots,M}, (q_m)_{m=0,1,\dots,M}), \\ q_m &= \frac{\log(F_m/S_0)}{t_m - t_0}, \quad m = 1, \dots, M. \end{aligned}$$

Algorithm. We divide the algorithm into steps.

Step 1 (Cost-of-carry rates). We compute the market cost-of-carry rates:

$$q_m = \frac{\log(F_m/S_0)}{t_m - t_0}, \quad m = 1, \dots, M.$$

Step 2 (Interpolation). We compute function

$$q(t) = \mathcal{I}((t_m)_{m=0,1,\dots,M}, (q_m)_{m=0,1,\dots,M}), \quad t \in [t_0, t_M].$$

Step 3. We return

$$F(t) = S_0 \exp(q(t)(t - t_0)), \quad t \in [t_0, t_M].$$

Least-squares fitting of data curves

While implementing the functions below, you need to account for the singularities of the type 0/0.

Discount curve obtained by fitting of yields

Input:

$(t_m)_{m=1,\dots,M}$: the maturities, $t_m < t_{m+1}$.

$(D_m)_{m=1,\dots,M}$: the discount factors.

t_0 : the initial time, $t_0 < t_1$.

$\mathcal{L}$: a fitting method for yields, a class `cfl::Fit`.

Output:

$D = D(t)$: the fitted discount curve.

$\epsilon = \epsilon(t)$: the error function of the fit for the discount curve.

$\mathbf{P} = (\widehat{\mathbf{c}}, \mathbf{\Gamma}, \chi^2)$: the parameters of the fit for the yield curve containing the fitted constants $\widehat{\mathbf{c}}$, the covariance matrix $\mathbf{\Gamma}$, and the total fitting error χ^2.

The discount curve has the form:

$$D(t) = \exp(-\gamma(t)(t - t_0)), \quad t \geq t_0,$$

where the yield curve $\gamma = \gamma(t)$ is the result of the $\mathcal{L}$-fit of the market yields:

$$\begin{aligned} \gamma(t) &= \mathcal{L}\left((t_m)_{m=1,\dots,M}, (\gamma_m)_{m=1,\dots,M}\right), \\ \gamma_m &= -\frac{\log D_m}{t_m - t_0}, \quad m = 1, \dots, M. \end{aligned}$$

Algorithm. We divide the algorithm into steps.

Step 1 (Compute the market yields). We have that

$$\gamma_m = -\frac{\log D_m}{t_m - t_0}, \quad m = 1, \dots, M.$$

Step 2 (Fit of the yields). We compute the least-squares fit for the yield curve:

$$\gamma(t) = \mathcal{L}((t_m)_{m=1,\dots,M}, (\gamma_m)_{m=1,\dots,M}).$$

We also compute the error function $\zeta = \zeta(t)$ of this fit and the fitting parameters $\mathbf{P}$.

Step 3 (Fit of the discount curve). We compute

$$D(t) = e^{-\gamma(t)(t-t_0)}, \quad \epsilon(t) = D(t)(t-t_0)\zeta(t), \quad t \geq t_0.$$

The formula for $\epsilon(t)$ holds by the first-order expansion with respect to $\zeta(t)$:

$$\epsilon(t) = |e^{-(\gamma(t)+\zeta(t))(t-t_0)} - D(t)| = D(t)|e^{-\zeta(t)(t-t_0)} - 1| \approx D(t)(t-t_0)\zeta(t).$$

Discount curve with constant yield obtained by the least-squares fitting of market yields

Input:

$(t_m)_{m=1,\dots,M}$: the maturities of the market discount factors, $t_m < t_{m+1}$.

$(D_m)_{m=1,\dots,M}$: the market discount factors.

t_0 : the initial time given as year fraction, $t_1 > t_0$.

Output:

$D = D(t)$: the fitted discount curve with constant yield.

$\epsilon = \epsilon(t)$: the error function of the fit for the discount curve.

$\mathbf{P} = (r, \Gamma, \chi^2)$: the fitting parameters for the yield curve containing the fitted constant yield, its variance, and the total fitting error.

The fitted discount curve has the form:

$$D(t) = \exp(-r(t-t_0)), \quad t \geq t_0,$$

where the constant yield r is the best least-squares fit to the market yields:

$$\chi^2 = \min_r \sum_{m=1}^{M} (r - \gamma_m)^2,$$
$$\gamma_m = -\frac{\log D_m}{t_m - t_0}, \quad m = 1, \dots, M.$$

Algorithm. We compute the discount curve $D = D(t)$ and its error function $\epsilon = \epsilon(t)$ by fitting the market yields with linear least-squares method and the constant basis function

$$f(t) = 1.$$

As part of the result, we obtain the fitting parameters $\mathbf{P} = (r, \Gamma, \chi^2)$ for the yield curve containing the fitted constant yield, its variance, and the total fitting error.

Discount curve obtained by the Nelson-Siegel fitting of market yields

Input:

$(t_m)_{m=1,\dots,M}$: the maturities, $t_m < t_{m+1}$.

$(D_m)_{m=1,\dots,M}$: the discount factors.

$\lambda > 0$: the mean-reversion rate.

t_0 : the initial time, $t_0 < t_1$.

Output:

$D = D(t)$: the fitted discount curve.

$\epsilon = \epsilon(t)$: the error function of the fit for the discount curve.

$\mathbf{P} = ((c_0, c_1, c_2), \Gamma, \chi^2)$: the fitting parameters for the yield containing the fitted constants, their covariance matrix, and the total fitting error.

The discount curve is given by

$$D(t) = \exp(-\gamma(t)(t - t_0)), \quad t \geq t_0,$$

where the yield curve $\gamma = \gamma(t)$ has the Nelson-Siegel form:

$$\begin{aligned} \gamma(t; c_0, c_1, c_2) &= c_0 + c_1 \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)} \\ &\quad + c_2 \left(\frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)} - e^{-\lambda(t-t_0)} \right), \quad t \geq t_0, \end{aligned}$$

and the constants c_0, c_1, c_2 are the result of the least-squares fit of the market yields:

$$\begin{aligned}
\chi^2 &= \min_{c_0,c_1,c_2} \sum_{m=1}^{M} (\gamma(t_m; c_0, c_1, c_2) - \gamma_m)^2, \\
\gamma_m &= -\frac{\log D_m}{t_m - t_0}, \quad m = 1, \dots, M.
\end{aligned}$$

Algorithm. We divide the algorithm into steps.

Step 1 (Basis functions). We compute the basis functions on $[t_0, \infty)$:

$$\begin{aligned}
f_1(t) &= 1, \\
f_2(t) &= \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)} = g_1(\alpha(t)), \\
f_3(t) &= \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)} - e^{-\lambda(t-t_0)} = g_2(\alpha(t)),
\end{aligned}$$

where

$$\begin{aligned}
\alpha(t) &= \lambda(t - t_0), \\
g_1(x) &= \frac{1}{x}\left(1 - e^{-x}\right), \\
g_2(x) &= g_1(x) - e^{-x}.
\end{aligned}$$

The functions g_1 and g_2 have a removable singularity at $x = 0$. They are implemented in `cfl` as `cfl::GetShape1` and `cfl::GetShape2`.

Step 2 (Fit of yield curve). We compute the discount curve $D = D(t)$ and its error function $\epsilon = \epsilon(t)$ by fitting the market yields with linear least-squares method and basis functions (f_1, f_2, f_3). As part of the result, we obtain the fitting parameters $\mathbf{P} = ((c_0, c_1, c_2), \Gamma, \chi^2)$ for the yield.

Discount curve for the Vasicek model obtained by the least-squares fitting of market yields

Input:

$(t_m)_{m=1,\dots,M}$: the maturities of market discount factors, $t_m < t_{m+1}$.

$(D_m)_{m=1,\dots,M}$: the market discount factors.

$\lambda > 0$: the mean reversion rate.

$\sigma > 0$: the short-term volatility.

t_0 : the initial time, $t_0 < t_1$.

Output:

$D = D(t)$: the fitted discount curve.

$\epsilon = \epsilon(t)$: the error function of the fit for the discount curve.

$\mathbf{P} = ((r_0, \theta), \Gamma, \chi^2)$: the fitted constants, their covariance matrix, and the total fitting error.

The discount curve is given by

$$D(t) = \exp(-\gamma(t)(t - t_0)), \quad t \geq t_0.$$

In the Vasicek model, the yield curve $\gamma = \gamma(t)$ has the form:

$$\gamma(t; r_0, \theta) = r_0 A(t) + \frac{\theta}{\lambda}(1 - A(t)) - \frac{\sigma^2}{2\lambda^2}(1 - 2A(t) + B(t)), \quad t \geq t_0,$$

where

$$A(t) = \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)}, \quad B(t) = \frac{1 - e^{-2\lambda(t-t_0)}}{2\lambda(t - t_0)}.$$

The initial short-term rate r_0 and the drift θ are the result of the least-squares fit of the market yields:

$$\chi^2 = \min_{r_0,\theta} \sum_{m=1}^{M} (\gamma(t_m; r_0, \theta) - \gamma_m)^2,$$

$$\gamma_m = -\frac{\log D_m}{t_m - t_0}, \quad m = 1, \dots, M.$$

Algorithm. We divide the algorithm into steps.

Step 1 (Basis functions). We compute the basis functions on $[t_0, \infty)$:

$$f_1(t) = A(t) = \frac{1 - e^{-\lambda(t-t_0)}}{\lambda(t - t_0)},$$

$$f_2(t) = \frac{1}{\lambda}(1 - A(t)),$$

and the free function

$$h(t) = -\frac{\sigma^2}{2\lambda^2}(1 - 2A(t) + B(t)), \quad t \geq t_0,$$

so that the yield curve $\gamma = \gamma(t)$ is given by

$$\gamma(t; r_0, \theta) = r_0 f_1(t) + \theta f_2(t) + h(t), \quad t \geq t_0.$$

We have that

$$\begin{aligned} f_1(t) &= g_1(\alpha(t)), \\ f_2(t) &= (t - t_0) g_3(\alpha(t)), \\ h(t) &= -\frac{1}{2}\sigma^2 (t - t_0)^2 g_4(\alpha(t)), \end{aligned}$$

where, for $t \geq t_0$ and $x \in \mathbb{R}$,

$$\begin{aligned} \alpha(t) &= \lambda(t - t_0), \\ g_1(x) &= \frac{1}{x}\left(1 - e^{-x}\right), \\ g_3(x) &= \frac{1}{x}\left(1 - g_1(x)\right), \\ g_4(x) &= \frac{1}{x^2}\left(1 - 2g_1(x) + g_1(2x)\right). \end{aligned}$$

The functions g_1, g_3, and g_4 have a removable singularity at $x = 0$. They are implemented in `cfl` as `cfl::GetShape1`, `cfl::GetShape3`, and `cfl::GetShape4`.

Step 2 (Fit of yield curve). We compute the discount curve $D = D(t)$ and its error function $\epsilon = \epsilon(t)$ by fitting the market yields with linear least-squares method based on the basis functions (f_1, f_2) and free function h. As part of the result, we obtain the fitting parameters $\mathbf{P} = ((r_0, \theta), \Gamma, \chi^2)$ for the yield.

Options on a stock

The issue time for all options coincides with the initial time. The maturities and coupon, barrier, and exercise times are strictly greater than the initial time.

Standard put

K : the strike.

T : the maturity.

The payoff of the option at the maturity is given by

$$V(T) = \max(K - S(T), 0),$$

where $S(T)$ is the price of the stock at T.

Algorithm. The event times are

$$\{t_0, T\},$$

where t_0 is the initial time. We have that

$$\begin{aligned} X(T) &= \max(K - S(T), 0), \\ X(t_0) &= \mathcal{R}_{t_0}(X(T)). \end{aligned}$$

At the end, we return $X(t_0)$.

American put

K : the strike.

$(t_m)_{m=1,\ldots,M}$: the exercise times.

A holder of the option can exercise it at any time t_m. In this case, he receives the intrinsic value

$$V(t_m) = \max(K - S(t_m), 0),$$

where $S(t_m)$ is the price of the stock at time t_m.

Algorithm. The event times are

$$\{t_0, \underbrace{(t_m)_{m=1,\ldots,M}}_{\text{exercise times}}\},$$

where t_0 is the initial time. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$X(t_M) = \underbrace{X(t_M)}_{>t_M} = 0.$$

Step 2 (Loop). We enter the loop at t_M (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_M}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1}}$$

is the value to continue (the exercises will be made after t_{m+1}). We have that

$$\underbrace{X(t_{m+1})}_{>t_m} = \max\Big(\underbrace{X(t_{m+1})}_{>t_{m+1}}, \underbrace{K - S(t_{m+1})}_{=t_m}\Big)$$

and then that

$$\underbrace{X(t_m)}_{>t_m} = \mathcal{R}_{t_m}\Big(\underbrace{X(t_{m+1})}_{>t_m}\Big).$$

Step 3 (After the loop). We return $X(t_0) = \underbrace{X(t_0)}_{>t_0}$.

Barrier up-or-down-and-out option

$U:$ the upper barrier.

$L:$ the lower barrier.

$(t_m)_{m=1,\dots,M}:$ the barrier times.

$N:$ the notional.

The payoff of the option at the maturity (the last barrier time t_M) is given by the notional amount N if the stock price stays between the lower and upper barriers for all barrier times. Otherwise, the option expires worthless.

Algorithm. The event times are

$$\{t_0, \underbrace{(t_m)_{m=1,\dots,M}}_{\text{barrier times}}\},$$

where t_0 is the initial time. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$X(t_M) = \underbrace{X(t_M)}_{>t_M} = N.$$

Step 2 (Loop). We enter the loop at t_M (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_M}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1}}$$

is the value to continue (no barriers were crossed before and at t_{m+1}). We have that

$$\underbrace{X(t_{m+1})}_{>t_m} = \underbrace{X(t_{m+1})}_{>t_{m+1}} 1_{\{S(t_{m+1})>L\}} 1_{\{U>S(t_{m+1})\}},$$

where $S(t)$ is the price of the stock at t, and then that

$$\underbrace{X(t_m)}_{>t_m} = \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_m}\big).$$

Step 3 (After the loop). We return $X(t_0) = \underbrace{X(t_0)}_{>t_0}$.

Forward on average spot price

$(t_m)_{m=1,\ldots,M}$: the averaging times.

At maturity t_M (the last averaging time), the holder of the long position in the forward contract receives the payment

$$V(t_M) = \frac{1}{M}\sum_{m=1}^{M} S(t_m) - G(t_0, t_M),$$

where $S(t)$ is the spot price at time t and $G(t_0, t_M)$ is the forward price computed at initial time t_0 for maturity t_M. We compute $G(t_0, t_M)$ in the general case of stochastic interest rates.

Algorithm. The event times are given by

$$\big\{t_0, \underbrace{(t_m)_{m=1,\ldots,M}}_{\text{averaging times}}\big\},$$

where t_0 is the initial time. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$X(t_M) = \underbrace{X(t_M)}_{>t_M} = 0.$$

Step 2 (Loop). We enter the loop at t_M (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_M}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$\underbrace{X(t_{m+1})}_{\text{known}} = \underbrace{X(t_{m+1})}_{>t_{m+1}}$$

is the value at t_{m+1} of the sum of the *future* spot prices paid at t_M:

$$\underbrace{X(t_{m+1})}_{>t_{m+1}} \triangleq \mathcal{R}_{t_{m+1},t_M}\big(\sum_{n=m+2}^{M} S(t_n)\big).$$

We have that

$$\begin{aligned}
\underbrace{X(t_{m+1})}_{>t_m} &\triangleq \mathcal{R}_{t_{m+1},t_M}\big(\sum_{n=m+1}^{M} S(t_n)\big)\\
&= \mathcal{R}_{t_{m+1},t_M}\big(\sum_{n=m+2}^{M} S(t_n)\big) + \mathcal{R}_{t_{m+1},t_M}\big(S(t_{m+1})\big)\\
&= \underbrace{X(t_{m+1})}_{>t_{m+1}} + S(t_{m+1})B(t_{m+1},t_M),
\end{aligned}$$

and then that

$$\underbrace{X(t_m)}_{>t_m} = \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_m}\big).$$

Step 3 (After the loop). We have that

$$X(t_0) = \underbrace{X(t_0)}_{>t_0} = \mathcal{R}_{t_0,t_M}\big(\sum_{m=1}^{M} S(t_m)\big).$$

We return

$$G(t_0,t_M) = \frac{X(t_0)}{MB(t_0,t_M)},$$

where we used the identities:

$$0 = \mathcal{R}_{t_0,t_M}\big(\frac{1}{M}\sum_{m=1}^{M} S(t_m) - G(t_0,t_M)\big) = \frac{1}{M}X(t_0) - G(t_0,t_M)B(t_0,t_M).$$

Forward on average spot price under deterministic interest rates

$(t_m)_{m=1,\dots,M}$: the averaging times.

At maturity t_M (the last averaging time), the holder of the long position in the forward contract receives the payment

$$V(t_M) = \frac{1}{M}\sum_{m=1}^{M} S(t_m) - G(t_0, t_M),$$

where $S(t)$ is the spot price at time t and $G(t_0, t_M)$ is the forward price computed at the initial time t_0 for maturity t_M. We assume that the interest rates are deterministic and compute $G(t_0, t_M)$ in a model with just one event time t_0.

Algorithm. The event times are given by the initial time only:

$$\{t_0\}.$$

We have that

$$\begin{aligned} X_m(t_0) &\triangleq \mathcal{R}_{t_0,t_M}\big(S(t_m)\big) \\ &= \mathcal{R}_{t_0,t_m}\big(\mathcal{R}_{t_m,t_M}\big(S(t_m)\big)\big) \\ &= \mathcal{R}_{t_0,t_m}\big(S(t_m)B(t_m,t_M)\big), \end{aligned}$$

where $B(t, t_M)$ is the discount factor at t for maturity t_M. As the interest rates are deterministic, the discount factors are also deterministic. Using this assumption, we obtain that

$$\begin{aligned} X_m(t_0) &= B(t_m,t_M)\mathcal{R}_{t_0,t_m}\big(S(t_m)\big) \\ &= B(t_m,t_M)\left(\mathcal{R}_{t_0,t_m}\big(S(t_m) - F(t_0,t_m)\big) + \mathcal{R}_{t_0,t_m}\big(F(t_0,t_m)\big)\right) \\ &= B(t_m,t_M)F(t_0,t_m)B(t_0,t_m) \\ &= F(t_0,t_m)B(t_0,t_M), \end{aligned}$$

where $F(t_0, t_m)$ is the forward price on the stock computed at t_0 for maturity t_m and we used the identity

$$B(t_0,t_M) = B(t_0,t_m)B(t_m,t_M),$$

which holds under the assumption of deterministic interest rates. It follows that

$$\begin{aligned}
0 &= \mathcal{R}_{t_0,t_M}\big(V(t_M)\big) = \mathcal{R}_{t_0,t_M}\big(\frac{1}{M}\sum_{m=1}^{M} S(t_m) - G(t_0,t_M)\big) \\
&= \frac{1}{M}\sum_{m=1}^{M} X_m(t_0) - \mathcal{R}_{t_0,t_M}\big(G(t_0,t_M)\big) \\
&= B(t_0,t_M)\left(\frac{1}{M}\sum_{m=1}^{M} F(t_0,t_m) - G(t_0,t_M)\right),
\end{aligned}$$

and then that

$$G(t_0,t_M) = \frac{1}{M}\sum_{m=1}^{M} F(t_0,t_m).$$

Down-and-out american call

$L:$ the lower barrier.

$(u_i)_{i=1,\dots,N_1}:$ the barrier times.

$K:$ the strike.

$(v_i)_{i=1,\dots,N_2}:$ the exercise times, $v_{N_2} > u_{N_1}$.

The option behaves as the american call with strike K and exercise times (v_i) until the first barrier time u_i when the stock price $S(u_i)$ is less than the lower barrier L. At this barrier time u_i, the option is canceled.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M}$ is the strictly increasing union of the barrier and exercise times. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$X(t_M) = \underbrace{X(t_M)}_{>t_M,>t_M} = 0.$$

Step 2 (Loop). We enter the loop at t_M (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_M}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}$$

is the value to continue (no barriers were crossed and exercises made before and at t_{m+1}). If t_{m+1} is both exercise and barrier time, then the barrier event takes precedence. There are two possibilities:

1. If t_{m+1} is an exercise time, then

$$\underbrace{X(t_{m+1})}_{>t_{m+1},>t_m} = \max\Big(\underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}, \underbrace{S(t_{m+1}) - K}_{>t_{m+1},=t_{m+1}}\Big),$$

   where $S(t)$ is the price of the stock at t.

2. If t_{m+1} is a barrier time, then

$$\underbrace{X(t_{m+1})}_{>t_m,>t_m} = \underbrace{X(t_{m+1})}_{>t_{m+1},>t_m} 1_{\{S(t_{m+1})>L\}}.$$

Finally, we have that

$$\underbrace{X(t_m)}_{>t_m,>t_m} = \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_m,>t_m}\big).$$

Step 3 (After the loop). We return $X(t_0) = \underbrace{X(t_0)}_{>t_0,>t_0}$.

Swing option

K : the strike.

$(t_n)_{n=1,\ldots,N}$: the exercise times.

M : the maximal number of exercises, $M \leq N$.

A holder of the option has the right to purchase M stocks at price K per share. The transactions take place at exercise times. Only *one* stock can be bought at a particular exercise time, that is, to get n stocks the holder should use n *different* exercise times. Such options are actively traded on energy markets.

Algorithm. The event times are

$$\{t_0, \underbrace{(t_n)_{n=1,\ldots,N}}_{\text{exercise times}}\},$$

where t_0 is the initial time. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$X_m(t_N) = \underbrace{X_m(t_N)}_{>t_N} = 0, \quad m = 0, 1, \ldots, M-1.$$

Step 2 (Loop). We enter the loop at t_N (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_N}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_n)}_{?} \longleftarrow \underbrace{X(t_{n+1})}_{\text{known}},$$

where

$$X(t_{n+1}) = (X_m(t_{n+1}))_{m=0,\ldots,M-1}$$

and

$$X_m(t_{n+1}) = \underbrace{X_m(t_{n+1})}_{>t_{n+1}}$$

is swing's value if m exercises were made before and at t_{n+1}. We have that

$$\underbrace{X_m(t_{n+1})}_{>t_n} = \max\Big(\underbrace{X_m(t_{n+1})}_{>t_{n+1}}, \underbrace{X_{m+1}(t_{n+1})}_{>t_{n+1}} + S(t_{n+1}) - K\Big),$$
$$m = 0, 1, \ldots, M-2,$$
$$\underbrace{X_{M-1}(t_{n+1})}_{>t_n} = \max\Big(\underbrace{X_{M-1}(t_{n+1})}_{>t_{n+1}}, S(t_{n+1}) - K\Big),$$

and then that

$$\underbrace{X_m(t_n)}_{>t_n} = \mathcal{R}_{t_n}\Big(\underbrace{X_m(t_{n+1})}_{>t_n}\Big), \quad m = 0, 1, \ldots, M-1.$$

Step 3 (After the loop). We return $X_0(t_0) = \underbrace{X_0(t_0)}_{>t_0}$.

Cross Currency Cap

$N:$ the notional.

$\delta R:$ the spread.

$\delta t:$ the time interval between the payments given as year fraction.

$M:$ the total number of payments.

We assume that today is the issue time of the cap and denote this time by t_0. The payment times are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \ldots, M.$$

At payment time t_m, the cap pays (in domestic currency)

$$N\delta t \max(r^{dom}(t_{m-1}, t_m) - r^{for}(t_{m-1}, t_m) - \delta R, 0),$$

where $r^{dom}(s,t)$ and $r^{for}(s,t)$ are the float rates computed at time s for maturity t for domestic and foreign currencies, respectively.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\ldots,M-1}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\ldots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps. We multiply on the notional at the end (Step 3).

Step 1 (Boundary condition).

$$\begin{aligned} X(t_{M-1}) &= \underbrace{X(t_{M-1})}_{>t_{M-1}} \\ &= \max\left(1 - B(t_{M-1}, t_{M-1}+\delta t)\delta R \delta t - \frac{S(t_{M-1})}{F(t_{M-1}, t_{M-1}+\delta t)}, 0\right), \end{aligned}$$

where the formula and the notations are explained on the next step.

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1}}$$

is the value to continue (the value of caplets paid after t_{m+1}). We have that

$$\underbrace{X(t_m)}_{>t_{m+1}} = \mathcal{R}_{t_m}\Big(\underbrace{X(t_{m+1})}_{>t_{m+1}}\Big)$$

and then that

$$\begin{aligned} \underbrace{X(t_m)}_{>t_m} &= \underbrace{X(t_m)}_{>t_{m+1}} + \\ &\quad B(t_m, t_{m+1})\delta t \max\left(r^{dom}(t_m, t_{m+1}) - \delta R - r^{for}(t_m, t_{m+1}), 0\right) \\ &= \underbrace{X(t_m)}_{>t_{m+1}} + \max\left(1 - B(t_m, t_{m+1})\delta R \delta t - \frac{S(t_m)}{F(t_m, t_{m+1})}, 0\right), \end{aligned}$$

where for times $u < v$ we denoted

$B(u, v):$ the domestic discount factor at u for maturity v,

$S(u):$ the spot FX at u,

$F(u, v):$ the forward FX at u for maturity v,

and used the identities:

$$\begin{aligned} B(u,v)(1 + r^{dom}(u,v)(v-u)) &= 1, \\ F(u,v)B(u,v)(1 + r^{for}(u,v)(v-u)) &= S_u. \end{aligned}$$

To derive the second identity, we start with the cash flows:

$$\begin{array}{ccc} S_u & \xrightarrow[\text{foreign bank}]{} & S_v(1 + r^{for}(u,v)(v-u)), \\ 0 & \xrightarrow[\text{long FX forward}]{} & S_v - F(u,v), \end{array}$$

and then deduce that

$$\begin{aligned} S_u &= \mathcal{R}_{u,v}\big(S_v(1 + r^{for}(u,v)(v-u))\big) \\ &= (1 + r^{for}(u,v)(v-u))\mathcal{R}_{u,v}\big(S_v\big) \\ &= (1 + r^{for}(u,v)(v-u))\mathcal{R}_{u,v}\big(F(u,v)\big) \\ &= (1 + r^{for}(u,v)(v-u))B(u,v)F(u,v). \end{aligned}$$

Step 3 (After the loop). We return $NX(t_0) = N\underbrace{X(t_0)}_{>t_0}$.

Options on interest rates

The issue time for all options coincides with the initial time. The maturities and coupon, barrier, and exercise times are strictly greater than the initial time.

Cap

N : the notional.

C : the cap rate.

δt : the interval of time between the payments given as year fraction.

M : the total number of payments.

We assume that today is the issue time of the cap and denote this time by t_0. The payment times of the cap are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M.$$

At payment time t_m, a holder receives the *caplet*

$$N \max(r(t_{m-1}, t_m)\delta t - C\delta t, 0),$$

where $r(s,t)$ is the float rate computed at time s for maturity t.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M-1}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps.

We multiply on the notional at the end (Step 3).

Step 1 (Boundary condition).

$$X(t_{M-1}) = \underbrace{X(t_{M-1})}_{>t_{M-1}} = \max(1 - B(t_{M-1}, t_M)(1 + C\delta t), 0).$$

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1}}$$

is the value to continue (the value of caplets paid after t_{m+1}). We have that

$$\underbrace{X(t_m)}_{>t_{m+1}} = \mathcal{R}_{t_m}\Big(\underbrace{X(t_{m+1})}_{>t_{m+1}}\Big)$$

and then that

$$\begin{aligned}\underbrace{X(t_m)}_{>t_m} &= \underbrace{X(t_m)}_{>t_{m+1}} + B(t_m, t_{m+1}) \max(r(t_m, t_{m+1})\delta t - C\delta t, 0) \\ &= \underbrace{X(t_m)}_{>t_{m+1}} + \max(1 - B(t_m, t_{m+1})(1 + C\delta t), 0),\end{aligned}$$

where $B(s, t)$ is the discount factor at s for maturity t and we used the identity:

$$B(s, t)(1 + r(s, t)(t - s)) = 1.$$

Step 3 (After the loop). We return $NX(t_0) = N\underbrace{X(t_0)}_{>t_0}$.

Swap

$N:$ the notional.

$R:$ the fixed rate.

$\delta t:$ the interval of time between the payments given as year fraction.

$M:$ the total number of payments.

side: the side of the swap contract, that is, whether one pays "fixed" and receives "float" or otherwise.

We assume that today is the issue date of the swap and denote this time by t_0. The payment times of the swap are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \ldots, M.$$

At time t_{m+1}, the side paying the float rate receives the amount

$$N\delta t(R - r(t_m, t_m + \delta t)),$$

where $r(s,t)$ is the float rate computed at s for maturity t. We compute the present value of the swap. We write the algorithm with just one event time t_0.

Remark. The same result holds for the swap on *compound rate* with K compounding intervals between payment times. In this contract, the side paying float receives at t_{m+1} the amount

$$N(1 + R\delta t - Y(t_m, t_{m+1}; K)),$$

where $Y(t_m, t_{m+1}; K)$ is the growth between t_m and t_{m+1} of the bank account paying market float rate for the period $\delta s = \delta t/K$:

$$Y(t_m, t_{m+1}; K) = \prod_{k=0}^{K-1} (1 + r(t_m + k\delta s, t_m + (k+1)\delta s)).$$

Algorithm. We need just one event time:

$$\{t_0\},$$

which is the initial time. Assume first that we pay float rate $r(s,t)$ and receive fixed rate R. Then

$$\underbrace{X(t_0)}_{\text{pay float}} = \underbrace{\text{swap}}_{\text{pay float}} = \text{coupon bond} - \text{bank account} = Y(t_0) - Z(t_0).$$

1. The coupon bond pays fixed coupons

   $$NR\delta t \quad \text{at} \quad t_m = t_0 + m\delta t, \quad m = 1, \ldots, M,$$

   and notional N at maturity t_M. Its value is given by

   $$Y(t_0) = N\left(R\delta t \sum_{m=1}^{M} B(t_0, t_m) + B(t_0, t_M)\right).$$

2. The bank account pays float interest

$$Nr(t_{m-1},t_m)\delta t \quad \text{at} \quad t_m = t_0 + m\delta t, \quad m = 1,\dots,M,$$

and notional N at maturity t_M. By replication, its value is given by the notional:

$$Z(t_0) = N.$$

If we pay fixed, then

$$\underbrace{X(t_0)}_{\text{pay fixed}} = -\underbrace{X(t_0)}_{\text{pay float}}.$$

Swaption

T : the maturity.

Parameters of underlying swap:

N : the notional.

R : the fixed rate.

δt : the interval of time between the payments given as year fraction.

M : the total number of payments.

side: the side of the swap contract, that is, whether we pay “fixed” and receive “float” or otherwise.

At maturity T, the holder has the right to enter into the swap contract with the parameters defined above and the issue time T.

Remark. The same result holds if at maturity T, the holder has the right to enter the swap on *compound rate* with K compounding intervals between payment times.

Algorithm. The event times have the form:

$$\{t_0, T\},$$

where t_0 is the initial time. We have that

$$X(T) = \max(V(T), 0),$$
$$X(t_0) = \mathcal{R}_{t_0}(X(T)),$$

where $V(T)$ is the value of the swap issued at T and $X(t)$ is the value of the swaption.

Cancellable collar

N : the notional.

C : the cap rate.

F : the floor rate ($F < C$).

δt : the interval of time between the payments given as year fraction.

M : the total number of payments.

We assume that today is the issue time of the collar and denote this time by t_0. The payment times of the collar are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M.$$

At payment time t_m, the following transactions take place:

1. If the float rate $r(t_{m-1}, t_m)$ is greater than cap rate C, then the holder *receives* the caplet:
$$N\delta t(r(t_{m-1}, t_m) - C).$$
2. If the float rate $r(t_{m-1}, t_m)$ is less than floor rate F, then the holder *pays* the flooret:
$$N\delta t(F - r(t_{m-1}, t_m)).$$
3. After the payment is either received or paid, the holder *has the right to cancel* the contract. No payments will be made after that.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M-1}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps.

We multiply on the notional at the end (Step 3).

Step 1 (Boundary condition).

$$X(t_{M-1}) = \underbrace{X(t_{M-1})}_{>t_{M-1},>t_{M-1}} = \max(1 - B(t_{M-1}, t_M)(1 + C\delta t), 0)$$
$$- \max(B(t_{M-1}, t_M)(1 + F\delta t) - 1, 0)$$

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1}, >t_{m+1}}$$

is the value to continue (the value of the collarets paid after t_{m+1} if we cancel after t_{m+1}). We have that

$$\underbrace{X(t_{m+1})}_{>t_{m+1}, >t_m} = \max\Big\{ \underbrace{X(t_{m+1})}_{>t_{m+1}, >t_{m+1}}, \underbrace{0}_{>t_{m+1}, =t_{m+1}} \Big\}$$

and then that

$$\underbrace{X(t_m)}_{>t_{m+1}, >t_m} = \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_{m+1}, >t_m}\big)$$

and

$$\begin{aligned}
\underbrace{X(t_m)}_{>t_m, >t_m} &= \underbrace{X(t_m)}_{>t_{m+1}, >t_m} + B(t_m, t_{m+1}) \max(r(t_m, t_{m+1})\delta t - C\delta t, 0) \\
&\quad - B(t_m, t_{m+1}) \max(F\delta t - r(t_m, t_{m+1})\delta t, 0) \\
&= \underbrace{X(t_m)}_{>t_{m+1}, >t_m} + \max(1 - B(t_m, t_{m+1})(1 + C\delta t), 0) \\
&\quad - \max(B(t_m, t_{m+1})(1 + F\delta t) - 1, 0),
\end{aligned}$$

where $B(s,t)$ is the discount factor at s for maturity t and we used the identity:

$$B(s,t)(1 + r(s,t)(t - s)) = 1.$$

Step 3 (After the loop). We return $NX(t_0) = N \underbrace{X(t_0)}_{>t_0, >t_0}$.

Down-and-out cap

Underlying cap:

> $N:$ the notional.
>
> $R:$ the cap rate.
>
> $\delta t:$ the interval of time between the payments given as year fraction.
>
> $M:$ the total number of payments.

$L:$ the lower bound for float rate.

We assume that today is the issue time of the cap and denote this time by t_0. The payment times of the cap are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M.$$

The down-and-out cap generates the same cash flow as the interest rate cap up to (and including) the payment time when the float rate drops below L. After that the option is terminated. In other words, if we denote by τ the first payment time t_m, when the float rate $r(t_m, t_m + \delta t)$ between t_m and $t_m + \delta t$ is less then L, then for a payment time t_j:

1. If $t_j \leq \tau$, then the holder gets the standard caplet

$$N \max(r(t_{j-1}, t_j)\delta t - R\delta t, 0).$$

2. If $t_j > \tau$, then the holder gets nothing.

Algorithm. The event times are

$$\left\{t_0, (t_m)_{m=1,\dots,M-1}\right\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps.

We multiply on the notional at the end (Step 3).

We denote by $B(s,t)$ and $r(s,t)$ the discount factor and the float rate, respectively, at time s for maturity t. We recall the identity:

$$B(s,t)(1 + r(s,t)(t-s)) = 1.$$

From this identity we deduce that

$$\begin{aligned} r(t,t+\delta t) > L &\iff 1 + r(t,t+\delta t)\delta t > 1 + L\delta t \\ &\iff 1 > B(t,t+\delta t)(1+L\delta t) \\ &\iff U > B(t,t+\delta t), \end{aligned}$$

where $U = 1/(1+L\delta t)$.

Step 1 (Boundary condition).

$$X(t_{M-1}) = \underbrace{X(t_{M-1})}_{>t_{M-1},>t_{M-1}} = \max(1 - B(t_{M-1},t_M)(1+C\delta t), 0).$$

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}$$

is the value to continue (the value of caplets paid after t_{m+1} if possible barrier events happen after t_{m+1}).

We have that

$$\begin{aligned} \underbrace{X(t_{m+1})}_{>t_{m+1},>t_m} &= \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}} 1_{\{r(t_{m+1},t_{m+1}+\delta t)>L\}} \\ &= \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}} 1_{\{U>B(t_{m+1},t_{m+1}+\delta t)\}} \end{aligned}$$

and then that

$$\underbrace{X(t_m)}_{>t_{m+1},>t_m} = \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_{m+1},>t_m}\big)$$

and

$$\begin{aligned}\underbrace{X(t_m)}_{>t_m,>t_m} &= \underbrace{X(t_m)}_{>t_{m+1},>t_m} + B(t_m, t_{m+1}) \max(r(t_m, t_{m+1})\delta t - C\delta t, 0) \\ &= \underbrace{X(t_m)}_{>t_{m+1},>t_m} + \max(1 - B(t_m, t_{m+1})(1 + C\delta t), 0),\end{aligned}$$

Step 3 (After the loop). We return $NX(t_0) = N \underbrace{X(t_0)}_{>t_0,>t_0}$.

Futures on float rate

The futures contracts of these types are traded, for example, on EUREX, where the underlying is 3 month EURO float rate.

Input: the parameters of futures contract.

Δ : the period for float rate given as year fraction ($\Delta = 0.25$ for 3 month float rate).

T : the maturity of futures contract.

M : the number of settlement times between today and the maturity.

Output: the futures price $F(t_0)$ computed at the initial time t_0.

We assume that the settlement times are given by

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M,$$

where t_0 is the initial time and

$$\delta t = \frac{T - t_0}{M}.$$

Notice that the settlement times include T, but do not contain t_0.

The futures contract on float rate involves the following transactions:

1. It costs nothing to take long or short position at any time.

2. At time t_m before the maturity, $m = 1, \dots, M - 1$,

(a) the buyer (long position) pays futures price $F(t_{m-1})$ established at the previous trading day,

(b) the seller (short position) pays futures price $F(t_m)$ established at the current trading day.

3. At maturity $T = t_M$,

   (a) the buyer (long position) pays futures price $F(t_{M-1})$ established at the previous trading day,

   (b) the seller (short position) pays

$$F(t_M) = F(T) = 1 - r(T, T + \Delta),$$

where $r(T, T+\Delta)$ is the float rate computed at time T for maturity $T + \Delta$.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M}$ are all futures times. We divide the algorithm into 3 steps.

Step 1 (Boundary condition).

$$F(t_M) = 1 - r(t_M, t_M + \Delta),$$

where

$$\begin{aligned} r(t_M, t_M + \Delta) &= \frac{1}{\Delta}\left((1 + r(t_M, t_M + \Delta)\Delta) - 1\right) \\ &= \frac{1}{\Delta}\left(\frac{1}{B(t_M, t_M + \Delta)} - 1\right) \end{aligned}$$

and $B(s, t)$ is the discount factor computed at s for maturity t.

Step 2 (Loop). We enter the loop at t_M (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_M}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{F(t_m)}_{?} \longleftarrow \underbrace{F(t_{m+1})}_{\text{known}}.$$

We have that

$$F(t_m) = \frac{1}{B(t_m, t_{m+1})} \mathcal{R}_{t_m, t_{m+1}} \left(F(t_{m+1}) \right),$$

because

$$0 = \mathcal{R}_{t_m, t_{m+1}} \left(F(t_{m+1}) - F(t_m) \right) = \mathcal{R}_{t_m, t_{m+1}} \left(F(t_{m+1}) \right) - B(t_m, t_{m+1}) F(t_m).$$

Step 3 (After the loop). We return $F(t_0)$.

Drop-lock swap

Brief description: a swap in which, the first time the market swap rate is above the upper barrier or below the lower barrier, the fixed rate is reset to the upper or lower barriers, respectively, and then remains constant.

Parameters of underlying swap:

N : the notional.

R : the initial fixed rate in the swap.

δt : the interval of time between payments given as year fraction.

M : the total number of payments.

side: the side of the swap contract. It defines whether the holder pays "fixed" rate and receives "float" rate or otherwise.

Reset rates:

L : the lower value for the swap rate after reset.

U : the upper value for the swap rate after reset, $L < U$.

We denote by $(t_m)_{m=1,\dots,M}$ the payment times of the swap:

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M,$$

and by $Q(t_m; \delta t, M)$ the market swap rate at t_m for the contract with the same period δt and the number of payments M as in the original swap. Let

$$\tau = \min\{(t_m)_{m=1,\dots,M} : \ Q(t_m; \delta t, M) > U \text{ or } Q(t_m; \delta t, M) < L\}.$$

Up to and including time τ, the payments in the swap contract are determined by initial fixed rate R. After τ, the fixed payments are given by U if $Q(\tau; \delta t, M) > U$ and by L if $Q(\tau; \delta t, M) < L$.

If neither upper nor lower barriers are crossed, then all payments are determined by original fixed rate R. Note that the initial time is *not* a reset time.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M-1}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps.

We denote

$Y(t,k,r):$ the value of interest rate swap issued at t with notional N, swap rate r, period δt, and the number of payments k. If we pay float and receive fixed payments, then

$$Y(t,k,r) = N\left(r\delta t \sum_{i=1}^{k} B(t, t+i\delta t) + B(t, t+k\delta t) - 1\right).$$

If we pay fixed, then we need to multiply this expression by -1.

$Q(t):$ the market swap rate computed at t for period δt and the number of payments M. We have that

$$Q(t) = \frac{1 - B(t, t+M\delta t)}{\delta t \sum_{m=1}^{M} B(t, t+m\delta t)}.$$

Step 1 (Boundary condition).

$$\underbrace{X(t_{M-1})}_{>t_{M-1}, >t_{M-1}} = Y(t_{M-1}, 1, R).$$

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}$$

is the value to continue (the value of payments after t_{m+1} if there were no resets before and at t_{m+1}). We have that

$$\begin{aligned}\underbrace{X(t_{m+1})}_{>t_{m+1},>t_m} = & \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}} \\ & + 1_{\{Q(t_{m+1})>U\}}\big(Y(t_{m+1}, M-(m+1), U) - \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}\big) \\ & + 1_{\{L>Q(t_{m+1})\}}\big(Y(t_{m+1}, M-(m+1), L) - \underbrace{X(t_{m+1})}_{>t_{m+1},>t_{m+1}}\big)\end{aligned}$$

and then that

$$\begin{aligned}\underbrace{X(t_m)}_{>t_{m+1},>t_m} &= \mathcal{R}_{t_m}\big(\underbrace{X(t_{m+1})}_{>t_{m+1},>t_m}\big), \\ \underbrace{X(t_m)}_{>t_m,>t_m} &= \underbrace{X(t_m)}_{>t_{m+1},>t_m} + Y(t_m, 1, R).\end{aligned}$$

Step 3 (After the loop). We return $\underbrace{X(t_0)}_{>t_0}$.

Auto cap

$t_0:$ the initial time.

Parameters of the cap:

$N:$ the notional.

$C:$ the cap rate.

$\delta t:$ the interval of time between the payments given as year fraction.

$M:$ the total number of periods.

$K:$ the maximal number of paid caplets, $K \leq M$.

The holder receives first K in-the-money caplets and then the contract is terminated. Recall that the caplet paid at time

$$t_m = t_0 + m\delta t, \quad m = 1, \dots, M,$$

is given by

$$N \max(r(t_{m-1}, t_m)\delta t - C\delta t, 0),$$

where $r(s,t)$ is the float rate computed at time s for maturity t.

Algorithm. The event times are

$$\{t_0, (t_m)_{m=1,\dots,M-1}\},$$

where t_0 is the initial time and $(t_m)_{m=1,\dots,M-1}$ are all payment times except the last one. We divide the algorithm into 3 steps.

We multiply on the notional at the end (Step 3).

For payment time t_m, we denote by $Y(t_m)$ the value of the next swaplet, where we pay the fixed rate C:

$$Y(t_m) = 1 - B(t_m, t_m + \delta t)(1 + C\delta t),$$

where $B(s,t)$ is the discount factor at s for maturity t and we used the identity:

$$B(s,t)(1 + r(s,t)(t-s)) = 1.$$

Step 1 (Boundary condition).

$$\underbrace{X_k(t_{M-1})}_{>t_{M-1}} = \max\left(Y(t_{M-1}), 0\right), \quad k = 0, 1, \dots, K-1.$$

Step 2 (Loop). We enter the loop at t_{M-1} (included) and exit at t_0 (not included):

$$\underbrace{t_0}_{\text{end}} \longleftarrow \underbrace{t_{M-1}}_{\text{begin}}.$$

We consider the iteration:

$$\underbrace{X(t_m)}_{?} \longleftarrow \underbrace{X(t_{m+1})}_{\text{known}},$$

where

$$X(t_{m+1}) = \underbrace{X_k(t_{m+1})}_{>t_{m+1}}, \quad k = 0, \dots, K-1,$$

and $\underbrace{X_k(t_{m+1})}_{>t_{m+1}}$ is the value to continue (the value of caplets paid after t_{m+1} if k caplets were in the money before and at t_{m+1}). We have that

$$\underbrace{X_k(t_m)}_{>t_{m+1}} = \mathcal{R}_{t_m}\Big(\underbrace{X_k(t_{m+1})}_{>t_{m+1}}\Big), \quad k = 0, \dots, K-1,$$

and then that

$$\begin{aligned}\underbrace{X_k(t_m)}_{>t_m} &= \underbrace{X_k(t_m)}_{>t_{m+1}} + 1_{\{Y(t_m)>0\}}\Big(\underbrace{X_{k+1}(t_m)}_{>t_{m+1}} + Y(t_m) - \underbrace{X_k(t_m)}_{>t_{m+1}}\Big),\\ &\qquad k = 0, \dots, K-2,\\ \underbrace{X_{K-1}(t_m)}_{>t_m} &= \underbrace{X_{K-1}(t_m)}_{>t_{m+1}} + 1_{\{Y(t_m)>0\}}\Big(Y(t_m) - \underbrace{X_{K-1}(t_m)}_{>t_{m+1}}\Big).\end{aligned}$$

Step 3 (After the loop). We return $N\,\underbrace{X_0(t_0)}_{>t_0}$.