Implementing some mathematical operators for a continuous-in-time financial model

Tarik Chakkour

Univ. Bretagne-Sud, UMR 6205, LMBA, F-56000 Vannes, France

Thursday 22nd December, 2016

Abstract. This paper considers the development of some mathematical operators as convolution and primitive for continuous-in-time financial model. This development is given in form of API (Application Programming Interface) with showing concept of its computation. The model is based on using measures and fields. The work we report here addresses the fundamental issue of how measures and fields are implemented for the software. The originality of this API lie in the fact that it will be used by the company MGDIS.

Keywords: API; convolution; primitive; discretization; integration; software tool.

1 Introduction

Time is the central element that influence financial economic behavior. The continuous-in-time financial model constitutes a powerful tool for studying the development of continuous-in-time methods in finance. We refer to papers [10, 9], which are dealing with continuous-in-time financial model. These papers develop the mathematics and economic theory of finance from the perspective of a model in which agents can revise their decisions continuously in time. At the same time, we have seen an explosion in the use of algorithms for computation methods to implement continuous-time models. The covered methods include convolution and primitive has been one of the most effective and widely-used of these methods. They began to be studied and applied systematically in various branches of modern science as in finance. We refer to [7] that presents an approach for implementing continuous-time adaptive recursive filters for convolution operator.

Within this paper, SOFI [1] is a software tool marketed by the company MGDIS. It is designed to the public institutions such local communities to set out multiyear budgets. SOFI is based on a discrete financial modeling. Currently, the mathematical objects involved in SOFI are suites and series. The discrete model generates outcomes in the form of tables. We showed in previous work [5] the default of this discrete model. We build a new model with using an other paradigm in [5]. This new model is based on continuous-in-time model and uses the mathematical tools such convolution and integration to describe loan scheme, reimbursement scheme and interest payment scheme. In [4, 3] we have shown some results about improving one of the continuous-in-time financial models built in paper [5]. We use in [3] a mathematical framework to discuss an inverse problem of the continuous-in-time model. We deal in [3] with its inverse problem stability.

This article describes implementing the continuous-in-time financial model. Mainly, we focus on concept of computation in API. This API is to be integrated in SOFI in order to produce the continuous software, and is restricted to certain measures and fields. The purpose of computing integration of measures over a time interval is to compute loan scheme, reimbursement scheme, etc; and the purpose of computing evaluation of fields at an instant is to compute current debt amount, where current debt field is a function that, at any time t, gives the capital amount still to be repaid.

Since some measures and fields could not been implemented continuously, we discretize them. Indeed, some computations in API need discretization. Next, we use these discrete values for obtaining the continuous values. The original motivation for this paper comes from a desire to understand the concept.
of computation in API with establishing mathematical relation between discrete measure and integrated measure. In addition, convolution and primitive operators are fundamental operations in the model. We use convolution in order to compute capital repayment measure with the Fast Fourier Transform method. We use primitive to compute current debt field at an instant \( t \) with accumulating measures between initial time and time \( t \). The primitive of measure is defined as a field in spite of it is undefined in the Radon measure space. In this work, we describe how we implement and check these operators.

The rest of this paper contains three sections. The first one introduces time steps that are involved in the models in order to show concept of computation in API. In the second we review numeric choices linked to API, where we define a field as continuous function by superior value. The last one shows implementation details about convolution and primitive.

## 2 Concept of computation in API

This section is devoted to explain time steps that are involved in the model and the relations between them. We give the time scales to integrate measure over interval which are shown in Figure 1. We introduce \( T_{\min} \) which is the time scale below which nothing coming from the model will be observed. To be more precise, we say that a measure \( \tilde{m} \) is observed over time interval \([t_1, t_2]\) if

\[
\int_{t_1}^{t_2} \tilde{m},
\]

is computed. And, we will always, choose times \( t_1 \) and \( t_2 \) such that \( t_2 - t_1 > T_{\min} \). In order to observe models, we need an observation step \( T_{\text{obs}} \) which is strictly superior to minimal observation step \( T_{\min} \)

\[
T_{\text{obs}} > T_{\min}.
\]

We define the discrete step \( T_{dM} \) as a smaller step than step \( T_{\min} \) to discretize measures:

\[
T_{dM} \leq T_{\min}.
\]

For instance, we are setting discrete step \( T_{dM} \) by following equality:

\[
T_{dM} = \frac{T_{\min}}{20}.
\]

Observation step \( T_{\text{obs}} \) is partitioned into \( n_D \) discrete step \( T_{dM} \) defined by:

\[
n_D = \left\lceil \frac{T_{\text{obs}}}{T_{dM}} \right\rceil.
\]

A field is evaluated between inferior value \( a \) and superior value \( b \) with discrete step \( T_{dF} \) satisfying:

\[
T_{dF} < b - a.
\]

Since measures and fields compose API, they are shared in two levels which are shown in Figure 2. First is high level and is created for business reasons. The computation in high level is designed to the SOFI users. Second is low level which is only used by the high level. The computation in low level is designed to the high level users. Notice that low level doesn’t use its high. We say that high level implements its low. High and low levels contain non-discrete measures, non-discrete fields defined on \( \mathbb{R} \), discrete measures and discrete fields. Some computations in high level need discretization. For instance, if we want to discretize a measure in high level, we create its copy in low level. Then we discretize it in order to rise up its values to high level.
The aim here is to explain how measures are integrated and how fields are evaluated. A non-discrete measure in low level is integrated between inferior bound $a$ and superior bound $b$ with minimal observation step $T_{\text{min}}$ and observation step $T_{\text{obs}}$. It follows that discrete step $T_{dM}$ is computed with relation (4). Whereas in high level it is integrated between inferior bounds $a$ and superior bound $b$. A non-discrete field in low level is evaluated between inferior value $a$ and superior value $b$ with discrete step $T_{dF}$. Yet, its evaluation in high level is done only between inferior value $a$ and superior value $b$.

The parallelism of discrete measures and fields in low level is based on the concept of a task. Tasks provide much benefits: more efficient computation and robustness API. Precisely, the Task Parallel Library [2] is used to entail execution and development speed. It is shown in [8] that this library makes it easy to take advantage of potential parallelism in a program. It relies heavily on generics and delegate expressions. Paper [13] shows several strategies that can be applied in large-scale discrete distribution clustering tasks.

In what follows, we build the unidimensional mesh called DAS (DiscretizedAxeSegment presented in Figure 3) for two reasons. First is to better structure the low level. Second is to compute discrete convolution due to the impossibility for computing it with variable step using the Fast Fourier Transform. Mesh DAS associated to discrete step $T_{dM}$ is defined by a set of points $(x_k)_{k \in \mathbb{Z}}$ that are its multiple

$$DAS_{T_{dM}} = \{x_k = k \times T_{dM}, k \in \mathbb{Z}\}. \quad (7)$$

Integration of measure $m_d$ in low level between inferior bound $a$ and superior bound $b$ with minimal observation step $T_{\text{min}}$ returns its integration between new inferior bound $x_a$ and new superior bound $x_b$ with discrete step $T_{dM}$, where

$$x_a = n_a \times T_{dM}, \quad (8)$$

such that:
\[ n_a = \left\lfloor \frac{a}{T_{dM}} \right\rfloor, \quad (9) \]

and where

\[ x_b = n_b \times T_{dM}, \quad (10) \]

such that:

\[
n_b = \begin{cases} 
\frac{b}{T_{dM}} & \text{if } T_{dM} \text{ is divisible by } b, \\
\left\lfloor \frac{b}{T_{dM}} \right\rfloor + 1 & \text{else.}
\end{cases} \quad (11)
\]

\[ n_b = n_a \]

Interval \([x_a, x_b]\) is partitioned into \(N_a^b\) subintervals of equal length, where \(N_a^b\) is given by:

\[ N_a^b = n_b - n_a, \quad (12) \]

where integers \(n_a\) and \(n_b\) are defined respectively in relations (9) and (11). Now, we will define a discrete measure of measure \(m_d\). For any integer \(j\) from 1 to \(N_a^b\), we call \((n_a + j - 1)^{nd}\) discrete value, the integration of measure \(m_d\) between inferior bound \((n_a + j - 1) \times T_{dM}\) and superior bound \((n_a + j) \times T_{dM}\) given by following equality:

\[
\forall j \in [1; N_a^b], m_d(n_a + j - 1) = \int_{(n_a + j - 1) \times T_{dM}}^{(n_a + j) \times T_{dM}} m_d.
\quad (13)
For any integer \( i \) from 1 to \( \left\lfloor \frac{N^b}{n_D} \right\rfloor \), we define quantity \( m^\text{obs}_d(i) \) as observed measure over time interval that its length is \( T^\text{obs} \) between inferior bound \( n_a \times T^\text{dM} + (i - 1) \times T^\text{obs} \) and superior bound \( n_a \times T^\text{dM} + i \times T^\text{obs} \). Formally, \( m^\text{obs}_d(i) \) is defined as:

\[
\forall i \in \left[ 1; \left\lfloor \frac{N^b}{n_D} \right\rfloor \right], m^\text{obs}_d(i) = \int_{n_a \times T^\text{dM} + (i - 1) \times T^\text{obs}}^{n_a \times T^\text{dM} + i \times T^\text{obs}} m_d, \tag{14}
\]

which is decomposed with Chasles relation as:

\[
\forall i \in \left[ 1; \left\lfloor \frac{N^b}{n_D} \right\rfloor \right], m^\text{obs}_d(i) = \sum_{k=1}^{n_D} \int_{(n_a + k - 1) \times T^\text{dM} + (i - 1) \times T^\text{obs}}^{(n_a + k) \times T^\text{dM} + i \times T^\text{obs}} m_d. \tag{15}
\]

Because of (5) and of the fact that \( l = k + (i - 1) \times n_D \), relation (15) implies that:

\[
\forall i \in \left[ 1; \left\lfloor \frac{N^b}{n_D} \right\rfloor \right], m^\text{obs}_d(i) = \sum_{l=1+(i-1) \times n_D}^{i \times n_D} \int_{(n_a + l - 1) \times T^\text{dM}}^{(n_a + l) \times T^\text{dM}} m_d. \tag{16}
\]

From this and according to (13), we conclude that observed value \( m^\text{obs}_d(i) \) is a sum of values \( m_d(n_a + l - 1) \) for integer \( l \) from \( 1 + (i - 1) \times n_D \) to \( i \times n_D \)

\[
\forall i \in \left[ 1; \left\lfloor \frac{N^b}{n_D} \right\rfloor \right], m^\text{obs}_d(i) = \sum_{l=1+(i-1) \times n_D}^{i \times n_D} m_d(n_a + l - 1). \tag{17}
\]

There are two situations for computing observed values. If \( N^b_a \) is divisible by \( n_D \), then \( \left\lfloor \frac{N^b}{n_D} \right\rfloor \) observed values are computed with relation (17). Else, \( \left\lfloor \frac{N^b}{n_D} \right\rfloor \) observed values are computed with relation (17) such that the observed value \( m^\text{obs}_d\left(\left\lfloor \frac{N^b}{n_D} \right\rfloor + 1\right) \) is computed with following relation:

\[
m^\text{obs}_d\left(\left\lfloor \frac{N^b}{n_D} \right\rfloor + 1\right) = \sum_{k=n_D \times \left\lfloor \frac{N^b}{n_D} \right\rfloor + 1}^{N^b_a} m_d(n_a + k - 1). \tag{18}
\]

### 3 Numeric choices in API

We are concerned in this section about implementation choices providing for great flexibility in API. Given a continuous function \( \phi \), the Dirac measure \( \delta_p \) at point \( p \) acts on the function \( \phi \). The value of this action is \( \phi(p) \). The purpose is to maintain this action in API. For that, we will explain the numeric choices that we have made to achieve it due to the difficulty for describing the dual of vector space of continuous piecewise function with a finite number of pieces, continuous with superior values. For instance, the action of Dirac measure \( \delta_p \) on fields \( \mathbb{1}_{[-\infty, p]} \) and \( \mathbb{1}_{[p, +\infty]} \) is undefined. Indeed, they integrals with respect to Dirac measure \( \delta_p \) could not be computed. Formally, following integrals

\[
\int_{-\infty}^{+\infty} \mathbb{1}_{[-\infty, p]} \, d\delta_p(x), \int_{-\infty}^{+\infty} \mathbb{1}_{[p, +\infty]} \, d\delta_p(x), \tag{19}
\]
are undefined. In order to set the value of this action consistently, we make a choice on Dirac measure \( \delta_p \) defined by:

\[
< \delta_p, \phi > = \lim_{x \to p^+} \phi(x).
\]  

Consider a continuous function \( g \) with integral equals 1 over \( \mathbb{R} \). To justify relation (20), we may restrict to support of function \( g \) defining function \( g_\epsilon \) which approaches Dirac measure \( \delta_p \), and is defined as:

\[
g_\epsilon(x) = \frac{1}{\epsilon^2} g\left(\frac{x - p}{\epsilon} + \epsilon p\right).
\]

(21)

Figure 4: Restriction to support of function \( g \) defining function \( g_\epsilon \) in relation (21).

Dirac measure \( \delta_p \) can be expressed as a limit of function \( g_\epsilon \)

\[
\lim_{\epsilon \to 0^+} g_\epsilon = \delta_p.
\]

(22)

To obtain relation (20), we require the following inclusion:

\[
\text{Supp}(g_\epsilon) \subset ]p, +\infty[.
\]

(23)

because of:

\[
\text{Supp}(g) \subset ]p, +\infty[.
\]

(24)

Relation (24) provides restriction to support of function \( g \) illustrated in Figure 4 due to following equivalence:

\[
\forall \epsilon \in \mathbb{R}^+_+, \ x > p \iff \frac{1}{\epsilon} \times \left(\frac{x - p}{\epsilon} + \epsilon p\right) > p.
\]

(25)

Using relation (20), the value of Dirac measure’s action \( \delta_p \) on field \( I_{]-\infty, p]} \) is 0, and the value of Dirac measure’s action \( \delta_p \) on field \( I_{[p, +\infty[} \) is 1. Consequently, the numeric choice given by relation (20) is consistent to make Dirac measure \( \delta_p \) act on a set of fields. Next, we can extend the restriction for any measure \( m_\epsilon \) converging to \( m \) as \( \epsilon \) approaches 0. This restriction to support of measure \( m_\epsilon \) is given by:

\[
\text{Supp}(m_\epsilon) \subset [\sup(\text{Supp}(m)), +\infty[.
\]

(26)
4 Convolution and accumulation

This section covers the implementation of convolution and of primitive operators. We refer to papers
[6, 12, 11], which are dealing with how convolution can be efficiently computed by FFT (the Fast Fourier
Transform). For example, algorithms based on explicit computation and on FFT are described in [6].
Paper [12] presents a more efficient computation of the convolution between a compressed text and an
uncompressed pattern. Schaller & Temnov estimates in [11] numerical errors of discrete FFT. In the
model, Loan Measure \( \hat{\kappa}_E \) is defined such that the amount borrowed between times \( t_1 \) and \( t_2 \) is:

\[
\int_{t_1}^{t_2} \hat{\kappa}_E.
\]  

(27)

and Repayment Measure \( \tilde{\rho}_C \) is defined such that the amount borrowed between times \( t_1 \) and \( t_2 \) is:

\[
\int_{t_1}^{t_2} \tilde{\rho}_C.
\]  

(28)

Loan Measure \( \hat{\kappa}_E \) and Capital Repayment Measure \( \tilde{\rho}_C \) are connected by a convolution operator. It is
required to implement it in order to compute repayment amount. Then the discrete convolution may
be evaluated with the aid of FFT method. By the Fourier convolution theorem, the discrete Fourier
transform of \( \hat{\kappa}_E \ast \tilde{\gamma} \) may be computed as

\[
\mathcal{F}(\tilde{\rho}_C) = \mathcal{F}(\hat{\kappa}_E \ast \tilde{\gamma}) = \mathcal{F}(\hat{\kappa}_E) \ast \mathcal{F}(\tilde{\gamma}),
\]  

(29)

where the Repayment Pattern Measure \( \tilde{\gamma} \) expresses the way an amount 1 borrowed at \( t = 0 \) is repaid and
where \( \ast \) denotes component-wise multiplication. Quantities \( \mathcal{F}(\hat{\kappa}_E) \) and \( \mathcal{F}(\tilde{\gamma}) \) define discrete Fourier transforms of \( \hat{\kappa}_E \) and of \( \tilde{\gamma} \), respectively. The computation of discrete convolution \( (\hat{\kappa}_E \ast \tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N^d} \)
with discrete measures \( (\hat{\kappa}_E(n_a + j - 1))_{1 \leq j \leq N^d} \) and \( (\tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N^d} \) between points \( x_e \) and \( x_f \) of
univercial mesh \( \text{DAST}_{\text{dM}} \) is summarized as follows:

- Determine the convex hull of the support of discrete measure \( (\hat{\kappa}_E(n_a + j - 1))_{1 \leq j \leq N^d} \) supposed to
  be interval \([x_{a_1}, x_{b_1}]\);
- Determine the convex hull of the support of discrete measure \( (\tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N^d} \) supposed to
  be interval \([x_{c_1}, x_{d_1}]\);
- Complete by zero discrete measures \( (\hat{\kappa}_E(n_a + j - 1))_{1 \leq j \leq N^d} \) and \( (\tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N^d} \) such that they have \( N \) values, where \( N \) is power of 2 and is smallest value satisfying \( N \geq N^b + N^d \). Then,
  \( (\hat{\kappa}_E(n_a + j - 1))_{1 \leq j \leq N} \) and \( (\tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N} \) are called the discrete values extended by zero;
- Compute discrete measures \( (x(n_a + j - 1))_{1 \leq j \leq N} \) and \( (y(n_c + j - 1))_{1 \leq j \leq N} \) by Fourier transform of
discrete measures \( (\hat{\kappa}_E(n_a + j - 1))_{1 \leq j \leq N} \) and \( (\tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N} \), respectively;
- Compute vector \( z(j - 1)_{1 \leq j \leq N} \) defined by element-wise multiplication of \( (x(n_a + j - 1))_{1 \leq j \leq N} \) by
  \( (y(n_c + j - 1))_{1 \leq j \leq N} \);
- Compute vector \( h(j - 1)_{1 \leq j \leq N} \) defined by inverse Fourier transform of \( (z(j - 1))_{1 \leq j \leq N} \);
- Construct tabulated measure \( \tilde{m}_{\text{Tabulated}} \) between inferior value \( x_{a_1} + x_{c_1} \) and superior value \( x_{b_1} + x_{d_1} \)
  with a set of first \( N^b + N^d \) values of \( h \);
- Discretize tabulated measure \( \tilde{m}_{\text{Tabulated}} \) between points \( x_e \) et \( x_f \) with discrete step \( T_{\text{dM}} \) to get
  discrete values \( (\hat{\kappa}_E \ast \tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N^d} \).
The integration of discrete measure $(\tilde{\kappa}_E \ast \tilde{\gamma}(n_c + j - 1))_{1 \leq j \leq N_a}$ in high level between inferior bound $e$ and superior bound $f$ is the sum of its values given by:

$$\sum_{j=1}^{N_a} \tilde{\kappa}_E \ast \tilde{\gamma}(n_c + j - 1).$$

After defining convolution operator, we describe how the accumulation of measure is defined in API. The Current Debt Field $K_{RD}$ is related to Loan Measure $\tilde{\kappa}_E$ and Repayment Measure $\tilde{RK}$ by the following Ordinary Differential Equation:

$$\frac{dK_{RD}}{dt} = \kappa_E(t) - R(t).$$

The solution of this ODE is expressed:

$$K_{RD}(t) = K_{RD}(t_1) + \int_{t_1}^{t} \tilde{\kappa}_E - \int_{t_1}^{t} \tilde{RK}.$$

To compute the Current Debt Field $K_{RD}$ at an instant $t$, we define the method that is computing the primitive of a measure. This method is based on numerical approach which consists in accumulating a discrete measure in order to approximate it by a field. The primitive of measure $m$ in low level that is zero at point $x_c$ is a field $F_d$. Its discretization between inferior value $x_a$ and superior value $x_b$ with discrete step $T_{df}$ is defined by discrete field $(F_d^P (n_a + k - 1))_{1 \leq k \leq N_a}$ given by:

$$\forall k \in [1; N_a + 1], F_d^P (n_a + k - 1) = \int_{x_c}^{y_k} m_d,$$

where points $(y_k)_{1 \leq k \leq N_a}$ are defined as:

$$\forall k \in [1; N_a + 1], y_k = x_c + (k - 1) \times T_{df}.$$

We distinguish three cases of computing discrete field $(F_d^P (n_a + k - 1))_{1 \leq k \leq N_a + 1}$:

First case $x_c < x_a$

Measure $m_d$ is discretized between points $x_c$ and $x_b$ with discrete step $T_{df}$ to compute discrete measure $(m_d(n_c + j - 1))_{1 \leq j \leq N_a}$. The integral defined in relation (32) is decomposed with Chasles relation to get:

$$\forall k \in [1; N_a + 1], F_d^P (n_a + k - 1) = \sum_{j=1}^{N_a} \int_{x_c + (j - 1) \times T_{df}}^{x_c + j \times T_{df}} m_d + \sum_{j=1}^{N_a} \int_{x_a + (j - 1) \times T_{df}}^{x_a + j \times T_{df}} m_d.$$  (34)

Replacing $x_a$ by $x_c + N_a \times T_{df}$ in relation (34), we obtain the following equality:

$$\forall k \in [1; N_a + 1], F_d^P (n_a + k - 1) = \sum_{j=1}^{N_a} \int_{x_c + (j - 1) \times T_{df}}^{x_c + j \times T_{df}} m_d + \sum_{j=1}^{N_a} \int_{x_c + (j - 1 + N_a) \times T_{df}}^{x_c + (j + N_a) \times T_{df}} m_d.$$  (35)

From this and using relation (13) which defines discrete measure, we get:

$$\forall k \in [1; N_a + 1], F_d^P (n_a + k - 1) = \sum_{j=1}^{N_a} m_d(n_c + j - 1) + \sum_{j=1}^{N_a} m_d(n_c + j - 1).$$  (36)
Second case \( x_c > x_b \)

Measure \( m_d \) is discretized between points \( x_a \) and \( x_c \) with discrete step \( T_dF \) to compute discrete measure \((m_d(n_a + j - 1))_{1 \leq j \leq N_c^a} \). It follows that Chasles relation applied to relation (32) gives:

\[
\forall k \in [1; N_c^a + 1], F_d^D(n_a + k - 1) = -\sum_{j=k}^{N_c^a} \int_{x_a+(j-1)\times T_dF}^{x_a+j\times T_dF} m_d.
\]

This reduces to following form:

\[
\forall k \in [1; N_c^a + 1], F_d^D(n_a + k - 1) = -\sum_{j=k}^{N_c^a} m_d(n_a + j - 1).
\]

Third case \( x_a \leq x_c \leq x_b \)

Determining integer \( L \in [1; N_c^a] \) satisfying following inequalities:

\[
y_L < x_c \leq y_{L+1}.
\]

Since \( x_c > y_k \) for each integer \( k \) from 1 to \( L \), the result of second case implies that:

\[
\forall k \in [1; L], F_d^D(n_a + k - 1) = -\sum_{j=k}^{N_c^a} m_d(n_a + j - 1).
\]

Replacing \( x_c \) by \( x_a + N_c^a \times T_dF \) in relation (32), we employ Chasles relation to obtain:

\[
\forall k \in [L + 1; N_c^a + 1], F_d^D(n_a + k - 1) = \sum_{j=1+N_c^a}^{k-1} m_d(n_a + j - 1).
\]

**Conflict of Interests**

The authors declare that there is no conflict of interests.

**References**


