List of content

Fourier Transformation

Assuming that $\phi(u)$ is in L^1, the original function can be recovered from its Fourier transform by inversion:

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \mathrm{e}^{-iux} \phi(u) \mathrm{d}u.$$

Lemma (Inversion Lemma)

Let $\phi(u)$ be a characteristic function and $f(x)$ be a probability density function of some continuous variable X. Then we have:

$$f(x) = \frac{1}{\pi} \Re \left(\int_0^{\infty} \mathrm{e}^{-iux} \phi(u) \mathrm{d}u \right)$$

Proof

From Fourier inverse we have:

$$\begin{aligned} f(x) &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u \\ &= \frac{1}{2\pi}\left(\int_{-\infty}^{0} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u + \int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u\right), \end{aligned}$$

where the first integral on the RHS can be written as:

$$\int_{-\infty}^{0} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u = \int_{0}^{\infty} \mathrm{e}^{ivx}\phi(-v)\mathrm{d}v$$

By taking the conjugate of the first integral and using the fact that the conjugate of an exponent equals the exponent of the conjugate, we find:

$$\begin{aligned} \int_{0}^{\infty} \mathrm{e}^{ivx}\phi(-v)\mathrm{d}v &= \int_{0}^{\infty} \overline{\mathrm{e}^{-iux}\phi(u)}\mathrm{d}u \\ &= \overline{\int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u}. \end{aligned}$$

Proof

Therefore:

$$\begin{aligned} f(x) &= \frac{1}{2\pi}\left(\int_{-\infty}^{0} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u + \int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u\right) \\ &= \frac{1}{2\pi}\left(\int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u + \overline{\int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u}\right), \end{aligned}$$

As for any complex number $z \in \mathbb{C}$ we have $z + \bar{z} = 2\Re(z)$ therefore the density can be expressed as:

$$f(x) = \frac{1}{\pi}\Re\left(\int_{0}^{+\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u\right).$$

2D Grid Discretization

Now, suppose that we discretize the domain for x, and u into N grid points, then we consider the vectors $\mathbf{f}, \phi \in \mathbb{C}^N$:

$$\mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}, \phi = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{N-1} \\ \phi_N \end{pmatrix}. \tag{1}$$

Fourier Transform Derivations

- Let us consider the following representation:

$$\boxed{\int_0^{\infty} \mathrm{e}^{-iux}\phi(u)\mathrm{d}u =: \int_0^{\infty} \gamma(u)\mathrm{d}u.}$$

- We define a Trapezoidal integration over domain $[0, u_{max}]$, for which we have:

$$\begin{aligned}\int_0^{u_{max}} \gamma(u)\mathrm{d}u &\approx \frac{\Delta_u}{2}\left[\gamma(u_1) + 2\sum_{n=2}^{N-1}\gamma(u_n) + \gamma(u_N)\right] \\ &= \Delta_u\left[\sum_{n=2}^{N-1}\gamma(u_n) + \frac{1}{2}\left(\gamma(u_1) + \gamma(u_N)\right)\right].\end{aligned}$$

If we set

$$\begin{aligned}u_{max} &= N\Delta_u, \\ u_n &= (n-1)\Delta_u \\ x_k &= -b + \Delta_x(k-1),\end{aligned}$$

where: $k = 1, \ldots, N$ to be the grid in the x-domain.

Fourier Transform Derivations

- The constant b is a tuning parameter which can be freely chosen, typically it should be associated with a low quantile of distribution such that $\mathbb{P}[X < b] < \epsilon$.
- Now, for:

$$\boxed{\gamma(u) = \mathrm{e}^{-iux}\phi(u)}$$

we have (note that summation runs from $n = 1$ up to N):

$$\int_0^{u_{max}} \gamma(u)\mathrm{d}u \approx \Delta_u \left[\sum_{n=1}^{N} \mathrm{e}^{-i[(n-1)\Delta_u][-b+\Delta_x(k-1)]}\phi(u_n) - \frac{1}{2}\left[\mathrm{e}^{-ixu_1}\phi(u_1) + \mathrm{e}^{ixu_N}\phi(u_N)\right]\right.$$

which becomes:

$$\int_0^{u_{max}} \gamma(u)\mathrm{d}u \approx \Delta_u \left[\sum_{n=1}^{N} \mathrm{e}^{-i\Delta_x\Delta_u(n-1)(k-1)}\mathrm{e}^{i(n-1)b\Delta_u}\phi(u_n) - \frac{1}{2}\left[\mathrm{e}^{-ixu_1}\phi(u_1) + \mathrm{e}^{ixu_N}\phi(u\right.\right.$$

Fourier Transform Derivations

- If we set

$$\Delta_x \Delta_u = \frac{2\pi}{N},$$

we obtain

$$\int_0^{u_{max}} \gamma(u)\mathrm{d}u \approx \Delta_u \left[\sum_{n=1}^{N} \mathrm{e}^{-i\frac{2\pi}{N}(n-1)(k-1)} \mathrm{e}^{i(n-1)b\Delta_u} \phi(u_n) - \frac{1}{2}\left[\mathrm{e}^{-ixu_1}\phi(u_1) + \mathrm{e}^{ixu_N}\phi(u_N) \right. \right.$$

- Since $u_n = (n-1)\Delta_u$ we further have:

$$\int_0^{u_{max}} \gamma(u)\mathrm{d}u \approx \Delta_u \left[\underbrace{\sum_{n=1}^{N} \mathrm{e}^{-i\frac{2\pi}{N}(n-1)(k-1)}}_{FFT} \mathrm{e}^{ibu_n}\phi(u_n) - \frac{1}{2}\left[\mathrm{e}^{-ixu_1}\phi(u_1) + \mathrm{e}^{ixu_N}\phi(u_N) \right] \right].$$

FFT Implementation

- So finally we obtain:

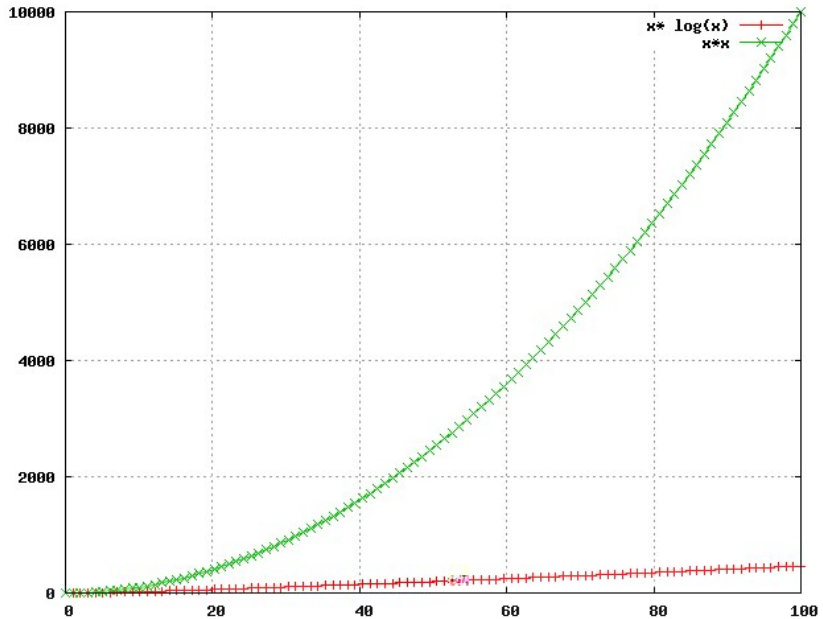
$$
\begin{aligned}
f(x) &= \frac{1}{\pi}\Re\left(\int_0^\infty \mathrm{e}^{-iux}\phi(u)\mathrm{d}u\right) \\
&= \frac{\Delta_u}{\pi}\Re\left\{\underbrace{\sum_{n=1}^{N}\mathrm{e}^{-i\frac{2\pi}{N}(n-1)(k-1)}}_{FFT}\mathrm{e}^{ibu_n}\phi(u_n)-\frac{1}{2}(\gamma_1+\gamma_2)\right\},
\end{aligned}
$$

  where $\gamma_1 = \mathrm{e}^{-ixu_1}\phi(u_1)$, and $\gamma_2 = \mathrm{e}^{ixu_N}\phi(u_N)$.

- Note that in the representation above we have considered a fixed grid for x_k, however we did not ensure that x in $f(x)$ is actually a point on that grid. For that reason, typically an interpolation between grid points of x_k is considered.

FFT Implementation

- This is a matrix multiplication, which requires about N^2 (complex) multiplications and N^2 (complex) additions. The number of arithmetic operations is of order N^2, i.e., $O(N^2)$.
- In 1965 Cooley and Tukey showed that it is possible to have the DFT evaluated in $O(N \log_2 N)$ operations.
- The algorithm was called the Fast Fourier Transform, FFT. Standard routines are available in many computer languages.



Again, we discretize the domain for x, and u into N grid points, and we consider the vectors $\mathbf{f}, \phi \in \mathbb{C}^N$:

$$\mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}, \phi = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{N-1} \\ \phi_N \end{pmatrix}. \tag{2}$$

Fourier Transformation

The discrete Fourier transform $\mathbf{f}$ of ϕ is given by the matrix multiplication:

$$\mathbf{f} = M\phi,$$

or equivalently:

$$f_k = \sum_{n=1}^{N} \phi_n \mathrm{e}^{-\frac{2\pi i}{N}(n-1)(k-1)} = \sum_{n=1}^{N} \phi_n \bar{\omega}_N^{(n-1)(k-1)}.$$

Fourier Transformation

If we let

$$\bar{\omega}_N = \mathrm{e}^{-\frac{2\pi i}{N}},$$

the discretized -Fourier Transform- matrix $M \in \mathbb{C}^{\mathbb{N}\times\mathbb{N}}$ is then defined as:

$$M = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \bar{\omega}_N^1 & \bar{\omega}_N^2 & \dots & \bar{\omega}_N^{N-1} \\ 1 & \bar{\omega}_N^2 & \bar{\omega}_N^4 & \dots & \bar{\omega}_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \bar{\omega}_N^{N-1} & \bar{\omega}_N^{N(N-1)} & \dots & \bar{\omega}_N^{(N-1)(N-1)} \end{pmatrix}, \tag{3}$$

that is,

$$M_{n,k} = \bar{\omega}_N^{(n-1)(k-1)}.$$

FFT

We explain the basis of the FFT, departing from:

$$f_k = \sum_{n=1}^{N} \phi_n \bar{\omega}_N^{(n-1)(k-1)}, \quad \text{for } k = 1, \ldots N.$$

with $N = 2^L$ (even).
Define $x_n = \phi_{2n-1}$ and $y_n = \phi_{2n}$ for $n = 1, \ldots N/2$ ($\{x_n\}$ and $\{y_n\}$ are the odd and even subsequences of $\{\phi_n\}$):

$$f_k = \sum_{n=1}^{N/2} (x_n \bar{\omega}_N^{(2n-2)(k-1)} + y_n \bar{\omega}_N^{(2n-1)(k-1)}), \quad \text{for } k = 1, \ldots N.$$

FFT

The basis behind the FFT is the simple but crucial equality:

$$\boxed{\bar{\omega}_N^{(2n-2)(k-1)} = \bar{\omega}_{N/2}^{(n-1)(k-1)}.}$$

With this equality, we have:

$$f_k = \sum_{n=1}^{N/2} x_n \bar{\omega}_{N/2}^{(n-1)(k-1)} + \bar{\omega}_N^{k-1} \sum_{n=1}^{N/2} y_n \bar{\omega}_{N/2}^{(n-1)(k-1)} = Z_k + \bar{\omega}_N^{k-1} Y_k$$

which is, apart from a factor, the sum of two DFTs, each of length $N/2$. Hence, we can write:

$$\begin{aligned} f_k &= Z_k + \bar{\omega}_N^{k-1} Y_k, \\ f_{k+N/2} &= Z_{k+N/2} + \bar{\omega}_N^{k-1+N/2} Y_{k+N/2}, \ k = 1, \ldots, N/2. \end{aligned}$$

FFT

Now noting that $Z_{k+N/2} = Z_k$, $Y_{k+N/2} = Y_k$ and $\bar{\omega}_N^{N/2} = -1$, this reduces to:

$$\begin{aligned} f_k &= Z_k + \bar{\omega}_N^{k-1} Y_k, \\ f_{k+N/2} &= Z_k - \bar{\omega}_N^{k-1} Y_k, \ k = 1, \ldots, N/2. \end{aligned}$$

the butterfly relations. They tell us how the DFT of a sequence can be recovered from the DFT of its odd and even subsequences. This procedure can be repeated using the sequences $\{x_n\}$ and $\{y_n\}$, until we have sequences of length 1. The resulting complexity is $O(N \log_2 N)$.

FFT Implementation

- Let us have a look at the FFT algorithm in Python
- In scipy.fft.fft we have:

Notes

FFT (Fast Fourier Transform) refers to a way the discrete Fourier Transform (DFT) can be calculated efficiently, by using symmetries in the calculated terms. The symmetry is highest when n is a power of 2, and the transform is therefore most efficient for these sizes. For poorly factorizable sizes, scipy.fft uses Bluestein's algorithm [2] and so is never worse than O(n log n). Further performance improvements may be seen by zero-padding the input using next_fast_len.

If `x` is a 1d array, then the fft is equivalent to

```
y[k] = np.sum(x * np.exp(-2j * np.pi * k * np.arange(n)/n))
```

The frequency term `f=k/n` is found at `y[k]`. At `y[n/2]` we reach the Nyquist frequency and wrap around to the negative-frequency terms. So, for an 8-point transform, the frequencies of the result are [0, 1, 2, 3, -4, -3, -2, -1]. To rearrange the fft output so that the zero-frequency component is centered, like [-4, -3, -2, -1, 0, 1, 2, 3], use fftshift.

Transforms can be done in single, double, or extended precision (long double) floating point. Half precision inputs will be converted to single precision and non-floating-point inputs will be converted to double precision.

If the data type of `x` is real, a "real FFT" algorithm is automatically used, which roughly halves the computation time. To increase efficiency a little further, use rfft, which does the same calculation, but only outputs half of the symmetrical spectrum. If the data are both real and symmetrical, the dct can again double the efficiency, by generating half of the spectrum from half of the signal.

When `overwrite_x=True` is specified, the memory referenced by `x` may be used by the implementation in any way. This may include reusing the memory for the result, but this is in no way guaranteed. You should not rely on the contents of `x` after the transform as this may change in future without warning.

The `workers` argument specifies the maximum number of parallel jobs to split the FFT computation into. This will execute independent 1-D FFTs within `x`. So, `x` must be at least 2-D and the non-transformed axes must be large enough to split into chunks. If `x` is too small, fewer jobs may be used than requested.

References

[1] Cooley, James W., and John W. Tukey, 1965, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.* 19: 297-301.

[2] Bluestein, L., 1970, "A linear filtering approach to the computation of discrete Fourier transform". *IEEE Transactions on*

FFT Implementation For CDF

- We take the characteristic function of the normal distribution $\mathcal{N}(\mu, \sigma^2)$:

$$\phi_X(u) = \exp\left(i\mu u - \frac{1}{2}\sigma^2 u^2\right)$$

  and we take $\mu = 0$, $\sigma = 1$.
- In the experiment we will use Python library scipy.fft.fft.
- Now, we compare the original PDF f_X, with the FFT approximation.

Pricing

How to get the price of a call option if the characteristic function of the asset is known?

- Gil-Palaez Inverse theorem,
- Carr-Madan Pricing,

$\Rightarrow$ COS Method

Fourier-Cosine Expansion

- The COS method:
  - Exponential convergence;
  - Greeks are obtained at no additional cost.
  - For discretely-monitored barrier and Bermudan options as well;
- The basic idea:
  - Replace the density by its Fourier-cosine series expansion;
  - Series coefficients have simple relation with characteristic function.

Fourier cosine expansions

The density and its characteristic function, $f_X(y)$ and $\phi_X(u)$, form a Fourier pair,

$$\phi_X(u) = \int_{\mathbb{R}} \mathrm{e}^{iyu} f_X(y) \mathrm{d}y, \tag{4}$$

and,

$$f_X(y) = \frac{1}{2\pi} \int_{\mathbb{R}} \mathrm{e}^{-iuy} \phi_X(u) \mathrm{d}u. \tag{5}$$

Fourier cosine expansions

- Fourier cosine series expansions give an optimal approximation of functions with a finite support.
- Definition of the Fourier expansion of function $g(x)$ on $[-1,1]$,

$$g(\theta) = {\sum}'_{k=0}^{\infty} \bar{A}_k \cos(k\pi\theta) + \sum_{k=1}^{\infty} \bar{B}_k \sin(k\pi\theta), \tag{6}$$

where the prime at the sum, ${\sum}'$, indicates that the first term in the summation is weighted by one-half, and the coefficients are given by:

$$\bar{A}_k = \int_{-1}^{1} g(\theta)\cos(k\pi\theta)\mathrm{d}\theta, \quad \bar{B}_k = \int_{-1}^{1} g(\theta)\sin(k\pi\theta)\mathrm{d}\theta. \tag{7}$$

Fourier cosine expansions

- By setting $\bar{B}_k = 0$, we obtain the classical Fourier cosine expansion, by which we can represent *even functions* around $\theta = 0$ exactly.
- We can extend any function $g : [0, \pi] \to \mathbb{R}$ to become an *even function on* $[-\pi, \pi]$, as follows,

$$\bar{g}(\theta) = \begin{cases} g(\theta), & \theta \geq 0 \\ g(-\theta), & \theta < 0. \end{cases} \tag{8}$$

- Even functions can be expressed as Fourier cosine series.

Fourier cosine expansions

- For a function $\bar{g}(\theta)$ supported on $[-\pi, \pi]$, the cosine expansion reads

$$\bar{g}(\theta) = \sum\nolimits'_{k=0}^{\infty} \bar{A}_k \cdot \cos(k\theta), \tag{9}$$

with

$$\bar{A}_k = \frac{1}{\pi} \int_{-\pi}^{\pi} \bar{g}(\theta) \cos(k\theta) \mathrm{d}\theta = \frac{2}{\pi} \int_{0}^{\pi} g(\theta) \cos(k\theta) \mathrm{d}\theta, \tag{10}$$

- Because $g(\theta) = \bar{g}(\theta)$ on $[0, \pi]$, the Fourier cosine expansion of the (not necessarily even) function $g(\theta)$ on $[0, \pi]$ is also given by the Equations (9), (10).
- The Fourier cosine series expansion, as used in the COS method, is based on a classical definition of the cosine series in the interval $[-\pi, \pi]$, with π being merely a scaling factor, and the function's maximum is attained at the domain boundary.

Fourier cosine expansions

- For functions supported on any other finite interval, say $[a, b] \in \mathbb{R}$, the Fourier cosine series expansion can be obtained via a *change of variables*:

$$\theta := \frac{y-a}{b-a}\pi, \quad y = \frac{b-a}{\pi}\theta + a.$$

- It then reads

$$g(y) = \sum\nolimits'_{k=0}^{\infty} \bar{A}_k \cdot \cos\left(k\pi \frac{y-a}{b-a}\right), \tag{11}$$

with

$$\bar{A}_k = \frac{2}{b-a}\int_a^b g(y)\cos\left(k\pi\frac{y-a}{b-a}\right)\mathrm{d}y. \tag{12}$$

Fourier cosine expansions

- Since any real function has a cosine expansion when it has finite support, the derivation of the approximation of a PDF starts with a truncation of the infinite integration range.
- Due to the conditions for the existence of a Fourier transform, the integrands in (5) have to decay to zero at $\pm\infty$ and we can truncate the integration range without losing significant accuracy.
- Suppose $[a, b] \in \mathbb{R}$ is chosen such that the *truncated integral* approximates the infinite counterpart very well, i.e.,

$$\hat{\phi}_X(u) := \int_a^b \mathrm{e}^{iuy} f_X(y)\mathrm{d}y \approx \int_{\mathbb{R}} \mathrm{e}^{iuy} f_X(y)\mathrm{d}y = \phi_X(u). \qquad (13)$$

- Relate (13) to (12), by recalling the Euler formula:

$$\boxed{\mathrm{e}^{iu} = \cos(u) + i\sin(u),}$$

which implies $\Re\{\mathrm{e}^{iu}\} = \cos(u)$, with $\Re\{\cdot\}$ the real part.

Fourier cosine expansions

- For random variable X and $a \in \mathbb{R}$,

$$\phi_X(u)\mathrm{e}^{ia} = \mathbb{E}[\mathrm{e}^{iuX+ia}] = \int_{-\infty}^{\infty} \mathrm{e}^{i(uy+a)} f_X(y)\mathrm{d}y. \qquad (14)$$

- By taking the real parts in (14),

$$\Re\left\{\phi_X(u)\mathrm{e}^{ia}\right\} = \Re\left\{\int_{-\infty}^{\infty} \mathrm{e}^{i(uy+a)} f_X(y)\mathrm{d}y\right\} = \int_{-\infty}^{\infty} \cos\left(uy+a\right) f_X(y)\mathrm{d}y.$$

- Substitute $u = \frac{k\pi}{b-a}$, and multiply (13) by $\exp\left(-i\frac{ka\pi}{b-a}\right)$, i.e.

$$\boxed{\hat{\phi}_X\left(\frac{k\pi}{b-a}\right)\cdot\exp\left(-i\frac{ka\pi}{b-a}\right) = \int_a^b \exp\left(iy\frac{k\pi}{b-a} - i\frac{ka\pi}{b-a}\right) f_X(y)\mathrm{d}y.}$$

- Taking the real part at both sides:

$$\boxed{\Re\left\{\hat{\phi}_X\left(\frac{k\pi}{b-a}\right)\cdot\exp\left(-i\frac{ka\pi}{b-a}\right)\right\} = \int_a^b f_X(y)\cos\left(k\pi\frac{y-a}{b-a}\right)\mathrm{d}y.} \quad (15)$$

Fourier cosine expansions

- It follows that $\bar{A}_k \approx \bar{F}_k$ with

$$\bar{F}_k := \frac{2}{b-a}\Re\left\{\phi_X\left(\frac{k\pi}{b-a}\right)\cdot\exp\left(-i\frac{ka\pi}{b-a}\right)\right\}. \tag{16}$$

- *Replace* $\bar{A}_k$ by $\bar{F}_k$ in the series expansion of $f_X(y)$ on $[a,b]$, i.e.,

$$\hat{f}_X(y) \approx {\sum}'_{k=0}^{\infty} \bar{F}_k \cos\left(k\pi\frac{y-a}{b-a}\right), \tag{17}$$

and *truncate the series summation*, so that

$$\boxed{\hat{f}_X(y) \approx {\sum}'_{k=0}^{N-1} \bar{F}_k \cos\left(k\pi\frac{y-a}{b-a}\right).} \tag{18}$$

- Remember that the first term in the summation should be multiplied by one-half (indicated by the $\sum'$ symbol).

Fourier cosine expansions

- Resulting error in approximation $\hat{f}_X(y)$ consists of a series truncation error and an error originating from approximation of $\bar{A}_k$ by $\bar{F}_k$.
- Since the cosine series expansions of so-called *entire functions* (i.e. functions without any singularities anywhere in the complex plane, except at ∞) exhibit an *exponential convergence*, we can expect (18) to give highly accurate approximations, with a small value for N, to density functions that have no singularities on $[a, b]$.

Summary on COS Method and Density Recovery

- Fourier-Cosine expansion of density function on interval $[a, b]$:

$$f(x) = \sum\nolimits'_{n=0}^{\infty} F_n \cos\left(n\pi \frac{x-a}{b-a}\right),$$

with $x \in [a, b] \subset \mathbb{R}$ and the coefficients defined as

$$F_n := \frac{2}{b-a} \int_a^b f(x) \cos\left(n\pi \frac{x-a}{b-a}\right) dx.$$

- F_n has direct relation to ch.f., $\phi(u) := \int_{\mathbb{R}} f(x) \mathrm{e}^{iux} dx$ ($\int_{\mathbb{R}\setminus[a,b]} f(x) \approx 0$),

$$\begin{aligned} F_n \approx A_n &:= \frac{2}{b-a} \int_{\mathbb{R}} f(x) \cos\left(n\pi \frac{x-a}{b-a}\right) dx \\ &= \frac{2}{b-a} \Re\left(\phi\left(\frac{n\pi}{b-a}\right) \exp\left(-i \frac{na\pi}{b-a}\right)\right). \end{aligned}$$

Recovering density

- PDF and characteristic function are known in closed-form, and read, respectively,

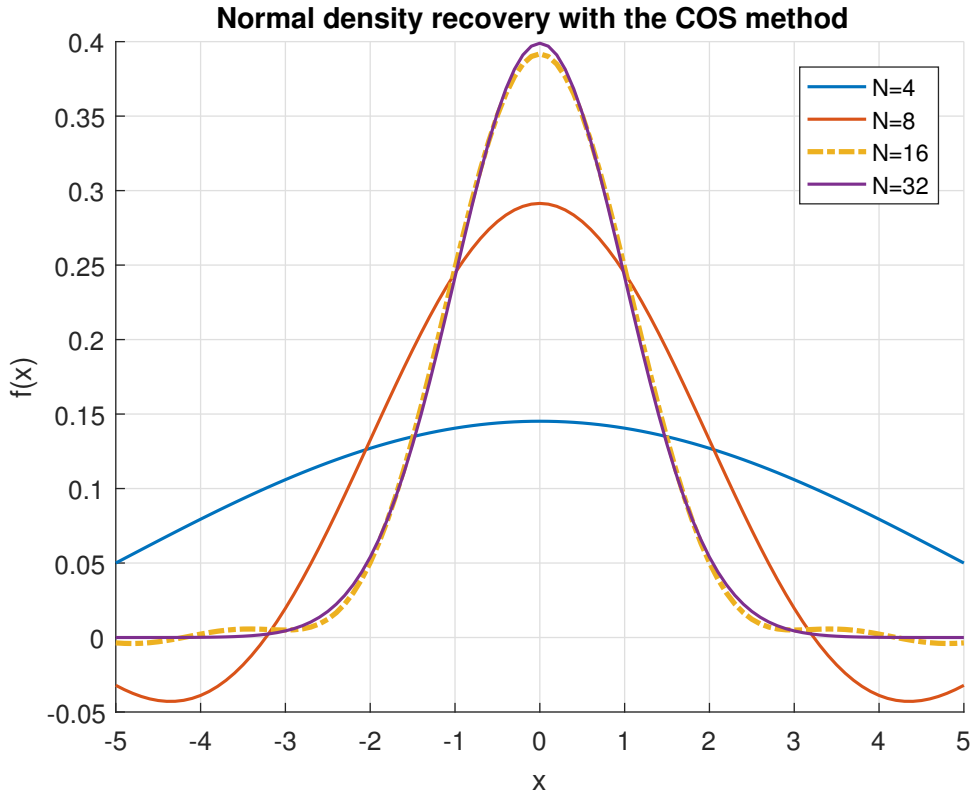
$$f_{\mathcal{N}(0,1)}(y) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}y^2}, \quad \phi_{\mathcal{N}(0,1)}(u) = \mathrm{e}^{-\frac{1}{2}u^2}.$$

- The accuracy of the expansion of the PDF for different N in (18). Integration interval $[a, b] = [-10, 10]$; maximum absolute error is measured at $y = \{-5, -4, \ldots, 4, 5\}$.
- From the differences in the CPU times in the table, "time(N)-time($N/2$)", observe a linear complexity.

N	4	8	16	32	64
Error	0.25	0.11	0.0072	4.04e-07	3.33e-17
CPU time (msec.)	0.046	0.061	0.088	0.16	0.29
Diff. in CPU (msec.)	–	0.015	0.027	0.072	0.13

Normal density recovery example

- The convergence of the normal density function.



- For small N the density is not well approximated (even with negative values), however, it converges exponentially.

Lognormal density approximation

- Employ the COS method to approximate the probability density function of a *lognormal random variable*.
- Characteristic function of lognormal RV is *not known*, and therefore we cannot simply apply the COS method.
- The CDF of Y in terms of the CDF of X can be calculated as,

$$F_Y(y) \overset{\text{def}}{=} \mathbb{P}[Y \leq y] = \mathbb{P}[\mathrm{e}^X \leq y] = \mathbb{P}[X \leq \log(y)] = F_X(\log(y)).$$

- By differentiation, we get:

$$f_Y(y) \overset{\text{def}}{=} \frac{\mathrm{d}F_Y(y)}{\mathrm{d}y} = \frac{\mathrm{d}F_X(\log(y))}{\mathrm{d}\log y}\frac{\mathrm{d}\log(y)}{\mathrm{d}y} = \frac{1}{y}f_X(\log(y)).$$

- Since X is normally distributed, and $\phi_X(u) = \mathrm{e}^{i\mu u - \frac{1}{2}\sigma^2 u^2}$, by the COS method, we may recover the PDF of a lognormal RV.

Lognormal density

- We set $\mu = 0.5$ and $\sigma = 0.2$ and analyze the convergence of the COS method, in dependence of the number of terms N.

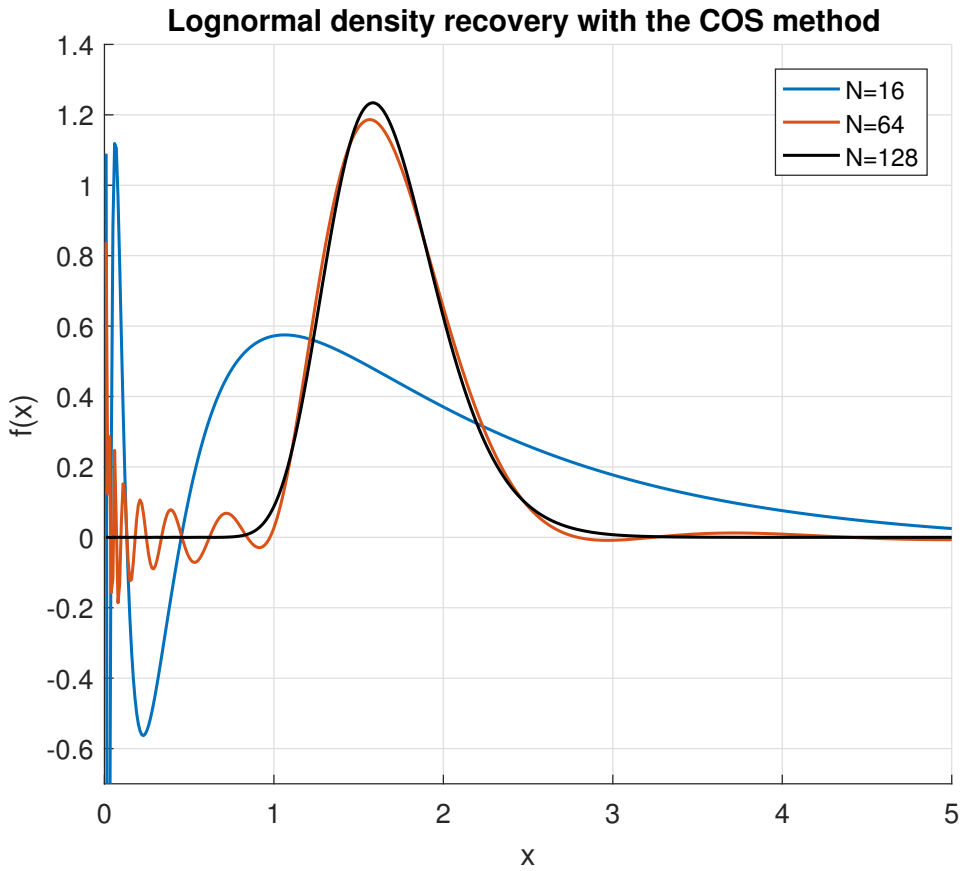


Figure: Recovery of the lognormal PDF for different numbers of terms.

Pricing European Options

- Start from the risk-neutral valuation formula:

$$V(x,t_0) = \mathrm{e}^{-r\Delta t}\mathbb{E}^{\mathbb{Q}}\left[V(y,T)|x\right] = \mathrm{e}^{-r\Delta t}\int_{\mathbb{R}} V(y,T)f(y|x)dy.$$

- Truncate the integration range:

$$V(x,t_0) = \mathrm{e}^{-r\Delta t}\int_{[a,b]} V(y,T)f(y|x)dy + \varepsilon.$$

- Replace the density by the COS approximation, and interchange summation and integration:

$$\hat{V}(x,t_0) = \mathrm{e}^{-r\Delta t}{\sum_{n=0}}'^{N-1}\Re\left(\phi\left(\frac{n\pi}{b-a};x\right)\mathrm{e}^{-in\pi\frac{a}{b-a}}\right)H_n,$$

where the series coefficients of the payoff, H_n, are analytic.

Payoff Coefficients

- The payoff for European options, in an *adjusted log-asset price*, i.e.

$$y(T) = \log\left(\frac{S(T)}{K}\right),$$

$$V(T,y) := [\bar{\alpha}\cdot K(\mathrm{e}^{y}-1)]^{+} \quad \text{with} \quad \bar{\alpha} = \begin{cases} 1 & \text{for a call,} \\ -1 & \text{for a put,} \end{cases}$$

- Focusing on a call option, *in the case that* $a < 0 < b$, we obtain

$$\begin{aligned} H_k^{\text{call}} &= \frac{2}{b-a}\int_0^b K(\mathrm{e}^{y}-1)\cos\left(k\pi\frac{y-a}{b-a}\right)\mathrm{d}y \\ &= \frac{2}{b-a}K\left(\chi_k(0,b)-\psi_k(0,b)\right), \end{aligned} \tag{19}$$

- Similarly, for a vanilla put, we find

$$H_k^{\text{put}} = \frac{2}{b-a}K\left(-\chi_k(a,0)+\psi_k(a,0)\right). \tag{20}$$

Integrals

- The cosine series coefficients, χ_k, of $g(y) = \mathrm{e}^y$ on an integration interval $[c, d] \subset [a, b]$,

$$\chi_k(c, d) := \int_c^d \mathrm{e}^y \cos\left(k\pi \frac{y-a}{b-a}\right) \mathrm{d}y, \tag{21}$$

and the cosine series coefficients, ψ_k, of $g(y) = 1$ on an integration interval $[c, d] \subset [a, b]$,

$$\psi_k(c, d) := \int_c^d \cos\left(k\pi \frac{y-a}{b-a}\right) \mathrm{d}y, \tag{22}$$

are known analytically.

Payoff Coefficients

- Basic calculus shows that

$$\chi_k(c,d) := \frac{1}{1+\left(\frac{k\pi}{b-a}\right)^2}\left[\cos\left(k\pi\frac{d-a}{b-a}\right)\mathrm{e}^d - \cos\left(k\pi\frac{c-a}{b-a}\right)\mathrm{e}^c + \frac{k\pi}{b-a}\sin\left(k\pi\frac{d-a}{b-a}\right)\mathrm{e}^d - \frac{k\pi}{b-a}\sin\left(k\pi\frac{c-a}{b-a}\right)\mathrm{e}^c\right],$$

and

$$\psi_k(c,d) := \begin{cases} \left[\sin\left(k\pi\frac{d-a}{b-a}\right) - \sin\left(k\pi\frac{c-a}{b-a}\right)\right]\frac{b-a}{k\pi}, & k \neq 0, \\ (d-c), & k = 0. \end{cases}$$

Multiple Strikes

- We can present the P_k as $\mathbf{P}_k = U_k \mathbf{K}$, where

$$U_k = \begin{cases} \frac{2}{b-a}\left(\bar{\chi}_k(0,b) - \bar{\psi}_k(0,b)\right) & \text{for a call} \\ \frac{2}{b-a}\left(-\bar{\chi}_k(a,0) + \bar{\psi}_k(a,0)\right) & \text{for a put.} \end{cases}$$

- The pricing formula simplifies for Heston and Lévy processes:

$$V(\mathbf{x}, t_0) \approx \mathbf{K} \mathrm{e}^{-r\Delta t} \cdot \Re\left(\sum\nolimits'_{n=0}^{N-1} \varphi\left(\frac{n\pi}{b-a}\right) U_n \cdot \mathrm{e}^{in\pi \frac{\mathbf{x}-a}{b-a}}\right),$$

where $\varphi(u) := \phi(u; 0)$

COS results, call and puts, GBM

- Parameters selected for this test are
  $S_0 = 100, \quad r = 0.1, \quad q = 0, \quad T = 0.1, \quad \sigma = 0.25.$
- The results for these strikes are obtained in one single computation.

N	16	32	64	128	256
msec.	0.10	0.11	0.13	0.15	0.19
abs. err. $K = 80$	3.67	7.44e-01	1.92e-02	1.31e-07	5.68e-14
abs. err. $K = 100$	3.87	5.28e-01	1.52e-02	3.87e-07	1.44e-13
abs. err. $K = 120$	3.17	8.13e-01	2.14e-02	3.50e-07	1.26e-13

- Linear computational complexity is achieved by the COS method.