E-companion to "Unified moment–based simulation of multivariate polynomial processes and applications in financial engineering"

Riccardo Brignone[a,*]

[a] *Department of Economics and Management, University of Pavia, Via San Felice 5, 27100 Pavia, Italy*

EC.1. Econometric performances of the Jacobi and Advanced Hull-White models: estimation under the historical measure

We follow the same approach as in Brignone and Gonzato (2024) to estimate the Heston, Jacobi and Advanced Hull-White models. The scope is to compare the econometric performances of the three models on real data. We consider the daily log-returns of the following assets: S&P500, EURO STOXX 50, USO and FTSE 100 ranging from April 10, 2006 to February 21, 2023 (4402 observations). The data source is Refinitiv Eikon (formerly, Thomson-Reuters' Datastream). The United States Oil (USO) fund is a popular exchange traded fund (ETF) that tracks the spot price of West Texas Intermediate (WTI). Statistical inference in stochastic volatility models is notoriously difficult because volatility is not directly observable and the models are highly nonlinear and non-Gaussian (preventing the use of standard Kalman filtering techniques). In line with Brignone and Gonzato (2024), we use Malik and Pitt (2011)'s continuous sequential importance resampling (CSIR) for parameter estimation. This algorithm addresses a common limitation of particle filters, which is that the likelihood is discontinuous with respect to unknown parameters, as illustrated by Doucet and Johansen (2008). The discontinuity in (simulated) likelihood makes the optimization problem extremely difficult. Malik and Pitt (2011) propose a continuous resampling strategy using a piecewise linear approximation for simulated maximum likelihood estimation (SMLE). To implement the CSIR we need to cast the Heston, Jacobi and Advanced Hull-White models in a state-space form. To this end we consider $X_t = \log(S_t/S_0)$ and an Euler approximation to the continuous-time system. Denoting by Δt a small time interval, we define (under the historical measure $\mathbb{P}$) the measurement as follows

$$X_t = \left(\mu - \frac{1}{2}V_{t-1}\right)\Delta t + \sqrt{V_{t-1}\Delta t}\, W_t, \tag{1}$$

while the transition equation is

$$\begin{cases} V_t = V_{t-1} + k(\theta - V_{t-1})\Delta t + \sigma\sqrt{V_{t-1}}\sqrt{\Delta t}Z_t, & \text{Heston} \\ V_t = V_{t-1} + k(\theta - V_{t-1})\Delta t + \sigma\sqrt{Q(V_{t-1})}\sqrt{\Delta t}Z_t, & \text{Jacobi} \\ Y_t = Y_{t-1} + k(\theta - Y_{t-1})\Delta t + \sigma Y_{t-1}\sqrt{\Delta t}Z_t, & \text{Advanced Hull-White} \end{cases}, \tag{2}$$

*Correspondence

Table 1: Priors specification.

Heston			Jacobi			Advanced Hull White		
Θ	Support	(μ_0, σ_0)	Θ	Support	(μ_0, σ_0)	Θ	Support	(μ_0, σ_0)
μ	$[-0.5, 0.5]$	$(0.01, 0.1)$	μ	$[-0.5, 0.5]$	$(0.01, 0.1)$	μ	$[-0.5, 0.5]$	$(0.01, 0.1)$
k	[0, 10]	(3, 3)	k	[0, 10]	(3, 3)	k	[0,10]	(3, 2)
θ	$[0, 0.7]$	$(0.04, 0.1)$	θ	$[0, 0.7]$	$(0.04, 0.1)$	θ	$[0, 0.1]$	$(0.2, 0.1)$
σ	$[0, 4]$	$(0.4, 1)$	σ	$[0, 4]$	$(0.4, 1)$	σ	$[0, 1]$	$(0.3, 0.2)$
ρ	$[-1, 1]$	$(-0.5, 0.5)$	ρ	$[-1, 1]$	$(-0.5, 0.5)$	ρ	$[-1, 1]$	$(-0.5, 0.5)$
–	–	–	v_{min}	$[0, 0.2]$	$(0.001, 0.01)$	–	–	–
–	–	–	v_{max}	$[0.2, 1.8]$	$(0.8, 0.1)$	–	–	–

Notes. Priors are truncated normal distributed where the truncation is defined by the support. The support is truncated to respect the domain of the parameters and to incorporate reasonable bounds that will be used in the optimization step.

where W_t and Z_t are correlated through parameter ρ, i.e. $Z_t = \rho W_t + \sqrt{1-\rho^2} B_t$, where W_t and B_t are independent standard normals, $Y_t = \sqrt{V_t}$ in the advanced Hull-White model. In order to alleviate the dependence on the initial guess, we implement the SMLE procedure in two steps: first, we simulate 480 sets of parameters from a truncated Gaussian prior with mean, variance, and support specified as in Table 1. For the Advanced Hull-White model, we set the maximum possible σ equal to 1. This is necessary in order to avoid the kurtosis explosion phenomenon noticed in Ackerer and Filipović (2020). We run the CSIR algorithm with 100 particles for each set of parameters and select the sets with the four highest log-likelihoods. Then, we run 4 optimizations on 500 particles, starting with the 4 parameter sets discovered in the previous step.

Table 2 displays the estimated parameters and standard errors. We use the same data and estimation procedure as in Brignone and Gonzato (2024), thus taking the same estimate for the Heston model. In unreported tests, we estimated the Heston model with the same setup and obtained very similar results. In addition to the parameter estimates, we provide the Bayesian Information Criterion (BIC). This enables us to compare the performances of the estimated models while also accounting for the fact that they have a different number of parameters. The BIC shows that the best performing models are Jacobi for S&P500, FTSE 100, and EURO STOXX 50, while the Hull-White model performs best for USO. The literature consistently shows that non-affine option pricing models outperform affine models (e.g., Heston in our case), see, among others, Fulop and Li (2019). This provides evidence for the utility of option pricing models based on polynomial processes. We emphasize that estimating the parameter v_{max} in the Jacobi stochastic volatility model is challenging, resulting in non-statistically significant estimates. Furthermore, the parameter σ in the Hull-White model is always roughly equal to 1, which is the upper bound we set to prevent kurtosis explosion. Removing the bound could increase the log-likelihood, but this would lead to the kurtosis explosion problem observed by Ackerer and Filipović (2020). In Section EC.11, we aim to improve the accuracy of our estimates and avoid such problems by including informative data from option prices.

Table 2: Estimated parameters for the Heston, Jacobi and Advanced Hull-White models on S&P500, FTSE100, EURO STOXX 50 and USO log-returns.

					Heston			
	S&P500		FTSE100		EURO STOXX 50		USO	
Θ	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.
μ	-0.1312	0.0570	0.0221	0.0561	-0.0329	0.0816	-0.0764	0.0389
k	6.9424	0.8693	2.2835	0.3868	5.1207	0.7877	4.9784	0.9256
θ	0.0418	0.0019	0.0561	0.0036	0.1192	0.0068	0.0264	0.0011
σ	0.7622	0.0343	0.5060	0.0338	1.0797	0.0513	0.4867	0.0265
ρ	-0.6746	0.0862	-0.4027	0.1468	-0.4531	0.0373	-0.7317	0.0957
BIC	-2.72E+04		-2.76E+04		-2.61E+04		-2.19E+04	

					Jacobi			
	S&P500		FTSE100		EURO STOXX 50		USO	
Θ	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.
μ	0.0149	0.0167	-0.0441	0.0282	-0.0381	0.0247	0.0265	0.0505
k	6.3796	0.0305	6.7938	0.1493	6.2415	0.4690	5.6090	0.3704
θ	0.0392	0.0016	0.0498	0.0033	0.0582	0.0041	0.1413	0.0081
σ	0.6774	0.0088	0.7583	0.0213	0.7816	0.0223	0.9227	0.0215
ρ	-0.7548	0.0084	-0.6589	0.0313	-0.7581	0.0217	-0.3952	0.0218
v_{min}	0.0006	0.0001	0.0026	0.0004	0.0023	0.0004	0.0006	0.0008
v_{max}	0.8392	0.6079	0.9775	1.2777	1.0318	0.5074	1.1417	1.0361
BIC	-2.90E+04		-2.88E+04		-2.70E+04		-2.21E+04	

					Advanced Hull-White			
	S&P500		FTSE100		EURO STOXX 50		USO	
Θ	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.	Estimate	s.e.
μ	0.0002	0.0281	0.0443	0.0269	0.0130	0.0338	-0.0135	0.0500
k	4.1222	0.2031	2.7413	0.2514	2.5421	0.1213	2.2206	0.0837
θ	0.1596	0.0054	0.1523	0.0082	0.2170	0.0106	0.4059	0.0203
σ	0.9988	0.0247	0.9946	0.0308	0.9984	0.0298	0.9994	0.0472
ρ	-0.5997	0.0339	-0.6793	0.0350	-0.8112	0.0247	-0.3727	0.0204
BIC	-2.88E+04		-2.88E+04		-2.69E+04		-2.22E+04	

Notes. s.e. denotes the Standard Error of the estimate.

EC.2. Matlab® codes for computing the joint moments

In this section, we provide a Matlab® code for computing the moments of $\left(V_T, \int_0^T V_s ds, X_T\right)$ in the Heston model. The code can be adapted to obtain the moments under the other models considered in the paper by simply modifying the variables `mu` and `sigma` (lines 12-13) of the supplied code.

```
1  NrMoments = 4;
2  model = 'Heston';
3  switch model
4      case 'Heston'
5          syms r k th sg rho
6          syms V L X % L denotes the integrated variance
7          assume(r>0 & k > 0 & th>0 & sg > 0)
8          assume(rho>-1 & rho<1)
9          assume(X>0 | X<0)
10         assume(L>0 & V>0)
11         mu = [k*(th-V) ; V ; r-1/2*V];
12         sigma = [sg*sqrt(V) 0; 0 0; rho*sqrt(V) sqrt(1-rho^2)*sqrt(V)];
13         D = 1/2*(sigma*sigma');
14 end
15 i = 0;
16 p_ = 1;
17 for N=1:NrMoments
18     for n3=0:N
19         for n2=0:N
20             for n1=0:N
21                 if n1+n2+n3≤N && n1+n2+n3>N-1
22                     f = V^n1*L^n2*X^n3;
23                     i=i+1;
24                     p_ = [p_; f];
25                     G = [diff(f,V); diff(f, L); diff(f,X)];
26                     H = [diff(f, V, 2) diff(diff(f, V),L) diff(diff(f, V),X);
27                         diff(diff(f, V),L) diff(f, L, 2) diff(diff(f, L),X);
28                         diff(diff(f, X),V) diff(diff(f, X),L) diff(f, X, 2)];
29                     val(i) = simplify(mu'*G + trace(H*D));
30                 end
31             end
32         end
33     end
34 end
35 val = [0 val];
36 for i=2:length(p_)
37     [cxy, txy] = coeffs(val(i), [V L X], 'All');
38     cxy2 = reshape(cxy, size(cxy,1)*size(cxy,2)*size(cxy,3), 1);
39     txy2 = reshape(txy, size(cxy,1)*size(cxy,2)*size(cxy,3), 1);
40     pos2 = find(cxy2≠0);
```

```
41      pos = zeros(length(pos2),1);
42      for jj=1:length(pos2)
43          [pos(jj)] = find(p_== txy2(pos2(jj)));
44      end
45      G(pos,i) = cxy2(pos2);
46  end
47  G = [G; zeros(length(p_)-size(G,1), size(G,2))];
48  out = matlabFunction(G, 'File', 'G_Heston.m',  "Vars", [k th sg rho r])
```

EC.3. Computing conditional moments given joint moments: examples in the special cases $d = 2$ and $d = 3$

In this section, we illustrate the procedure outlined in Section 2.3 for two relevant special cases, $d = 2$ and $d = 3$. Let us start considering the case $d = 2$ where we are interested in computing the moments of $(Z_2|Z_1)$. Define $\mu_{r_1,r_2} = \mu(r_1, r_2) = \mathrm{E}\left[Z_1^{r_1} Z_2^{r_2}\right]$, the system in (4) becomes $Ax = b$ where

$$A = \begin{bmatrix} 1 & \mu_{1,0} & 0 & 0 & 1 & 0 & \mu_{1,0} & 2\mu_{0,1} \\ \mu_{1,0} & \mu_{2,0} & 0 & 0 & \mu_{1,0} & 0 & \mu_{2,0} & 2\mu_{1,1} \\ \mu_{0,1} & \mu_{1,1} & 1 & \mu_{1,0} & 2\mu_{0,1} & \mu_{2,0} & 2\mu_{1,1} & 3\mu_{0,2} \\ \mu_{2,0} & \mu_{3,0} & 0 & 0 & \mu_{2,0} & 0 & \mu_{3,0} & 2\mu_{2,1} \\ \mu_{1,1} & \mu_{2,1} & \mu_{1,0} & \mu_{2,0} & 2\mu_{1,1} & \mu_{3,0} & 2\mu_{2,1} & 3\mu_{1,2} \\ \mu_{0,2} & \mu_{1,2} & 2\mu_{0,1} & 2\mu_{1,1} & 3\mu_{0,2} & 2\mu_{2,1} & 3\mu_{1,2} & 4\mu_{0,3} \\ \mu_{3,0} & \mu_{4,0} & 0 & 0 & \mu_{3,0} & 0 & \mu_{4,0} & 2\mu_{3,1} \\ \mu_{2,1} & \mu_{3,1} & \mu_{2,0} & \mu_{3,0} & 2\mu_{2,1} & \mu_{4,0} & 2\mu_{3,1} & 3\mu_{2,2} \\ \mu_{1,2} & \mu_{2,2} & 2\mu_{1,1} & 2\mu_{2,1} & 3\mu_{1,2} & 2\mu_{3,1} & 3\mu_{2,2} & 4\mu_{1,3} \\ \mu_{0,3} & \mu_{1,3} & 3\mu_{0,2} & 3\mu_{1,2} & 4\mu_{0,3} & 3\mu_{2,2} & 4\mu_{1,3} & 5\mu_{0,4} \end{bmatrix}, \quad b = -\begin{bmatrix} \mu_{0,1} \\ \mu_{1,1} \\ \mu_{0,2} \\ \mu_{2,1} \\ \mu_{1,2} \\ \mu_{0,3} \\ \mu_{3,1} \\ \mu_{2,2} \\ \mu_{1,3} \\ \mu_{0,4} \end{bmatrix},$$

where $x^T = \begin{bmatrix} a_0 & a_1 & b_0 & b_1 & b_2 & b_{11} & b_{12} & b_{22} \end{bmatrix}$. Given the entries of the vector x, (5) becomes

$$a_0^\star = a_0 + a_1 Z_1, \quad b_0^\star = -(b_0 + b_1 Z_1 + b_{11} Z_1^2), \quad b_1^\star = -(b_2 + b_{12} Z_1), \quad b_{11}^\star = -b_{22}.$$

Thus, conditional moments can be computed according to

$$\begin{aligned} \mathrm{E}[Z_2|Z_1] &= (b_1^\star - a_0^\star)(1 - 2b_{11}^\star)^{-1}, \\ \mathrm{E}[Z_2^2|Z_1] &= ((2b_1^\star - a_0^\star)\mathrm{E}[Z_2|Z_1] + b_0^\star)(1 - 3b_{11}^\star)^{-1}, \\ \mathrm{E}[Z_2^3|Z_1] &= ((3b_1^\star - a_0^\star)\mathrm{E}[Z_2^2|Z_1] + 2b_0^\star \mathrm{E}[Z_2|Z_1])(1 - 4b_{11}^\star)^{-1}, \\ \mathrm{E}[Z_2^4|Z_1] &= ((4b_1^\star - a_0^\star)\mathrm{E}[Z_2^3|Z_1] + 3b_0^\star \mathrm{E}[Z_2^2|Z_1])(1 - 5b_{11}^\star)^{-1}. \end{aligned}$$

Given conditional moments, it is possible to draw a random sample from $(Z_2|Z_1)$ through the Pearson's system of distributions.

We consider now the case $d = 3$. We are interested in computing the moments of $(Z_3|Z_2, Z_1)$. Define $\mu_{r_1,r_2,r_3} = \mu(r_1, r_2, r_3) = \mathrm{E}\left[Z_1^{r_1} Z_2^{r_2} Z_3^{r_3}\right]$. The system (4) becomes $Ax = b$ where:

$$A = \begin{bmatrix}
1 & \mu_{1,0,0} & \mu_{0,1,0} & 0 & 0 & 0 & 1 & 0 & 0 & \mu_{1,0,0} & 0 & \mu_{0,1,0} & 2\mu_{0,0,1} \\
\mu_{1,0,0} & \mu_{2,0,0} & \mu_{1,1,0} & 0 & 0 & 0 & \mu_{1,0,0} & 0 & 0 & \mu_{2,0,0} & 0 & \mu_{1,1,0} & 2\mu_{1,0,1} \\
\mu_{0,1,0} & \mu_{1,1,0} & \mu_{0,2,0} & 0 & 0 & 0 & \mu_{0,1,0} & 0 & 0 & \mu_{1,1,0} & 0 & \mu_{0,2,0} & 2\mu_{0,1,1} \\
\mu_{0,0,1} & \mu_{1,0,1} & \mu_{0,1,1} & 1 & \mu_{1,0,0} & \mu_{0,1,0} & 2\mu_{0,0,1} & \mu_{2,0,0} & \mu_{1,1,0} & 2\mu_{1,0,1} & \mu_{0,2,0} & 2\mu_{0,1,1} & 3\mu_{0,0,2} \\
\mu_{2,0,0} & \mu_{3,0,0} & \mu_{2,1,0} & 0 & 0 & 0 & \mu_{2,0,0} & 0 & 0 & \mu_{3,0,0} & 0 & \mu_{2,1,0} & 2\mu_{2,0,1} \\
\mu_{0,2,0} & \mu_{1,2,0} & \mu_{0,3,0} & 0 & 0 & 0 & \mu_{0,2,0} & 0 & 0 & \mu_{1,2,0} & 0 & \mu_{0,3,0} & 2\mu_{0,2,1} \\
\mu_{0,0,2} & \mu_{1,0,2} & \mu_{0,1,2} & 2\mu_{0,0,1} & 2\mu_{1,0,1} & 2\mu_{0,1,1} & 3\mu_{0,0,2} & 2\mu_{2,0,1} & 2\mu_{1,1,1} & 3\mu_{1,0,2} & 2\mu_{0,2,1} & 3\mu_{0,1,2} & 4\mu_{0,0,3} \\
\mu_{1,1,0} & \mu_{2,1,0} & \mu_{1,2,0} & 0 & 0 & 0 & \mu_{1,1,0} & 0 & 0 & \mu_{2,1,0} & 0 & \mu_{1,2,0} & 2\mu_{1,1,1} \\
\mu_{1,0,1} & \mu_{2,0,1} & \mu_{1,1,1} & \mu_{1,0,0} & \mu_{2,0,0} & \mu_{1,1,0} & 2\mu_{1,0,1} & \mu_{3,0,0} & \mu_{2,1,0} & 2\mu_{2,0,1} & \mu_{1,2,0} & 2\mu_{1,1,1} & 3\mu_{1,0,2} \\
\mu_{0,1,1} & \mu_{1,1,1} & \mu_{0,2,1} & \mu_{0,1,0} & \mu_{1,1,0} & \mu_{0,2,0} & 2\mu_{0,1,1} & \mu_{2,1,0} & \mu_{1,2,0} & 2\mu_{1,1,1} & \mu_{0,3,0} & 2\mu_{0,2,1} & 3\mu_{0,1,2} \\
\mu_{3,0,0} & \mu_{4,0,0} & \mu_{3,1,0} & 0 & 0 & 0 & \mu_{3,0,0} & 0 & 0 & \mu_{4,0,0} & 0 & \mu_{3,1,0} & 2\mu_{3,0,1} \\
\mu_{0,3,0} & \mu_{1,3,0} & \mu_{0,4,0} & 0 & 0 & 0 & \mu_{0,3,0} & 0 & 0 & \mu_{1,3,0} & 0 & \mu_{0,4,0} & 2\mu_{0,3,1} \\
\mu_{0,0,3} & \mu_{1,0,3} & \mu_{0,1,3} & 3\mu_{0,0,2} & 3\mu_{1,0,2} & 3\mu_{0,1,2} & 4\mu_{0,0,3} & 3\mu_{2,0,2} & 3\mu_{1,1,2} & 4\mu_{1,0,3} & 3\mu_{0,2,2} & 4\mu_{0,1,3} & 5\mu_{0,0,4} \\
\mu_{2,1,0} & \mu_{3,1,0} & \mu_{2,2,0} & 0 & 0 & 0 & \mu_{2,1,0} & 0 & 0 & \mu_{3,1,0} & 0 & \mu_{2,2,0} & 2\mu_{2,1,1} \\
\mu_{2,0,1} & \mu_{3,0,1} & \mu_{2,1,1} & \mu_{2,0,0} & \mu_{3,0,0} & \mu_{2,1,0} & 2\mu_{2,0,1} & \mu_{4,0,0} & \mu_{3,1,0} & 2\mu_{3,0,1} & \mu_{2,2,0} & 2\mu_{2,1,1} & 3\mu_{2,0,2} \\
\mu_{1,2,0} & \mu_{2,2,0} & \mu_{1,3,0} & 0 & 0 & 0 & \mu_{1,2,0} & 0 & 0 & \mu_{2,2,0} & 0 & \mu_{1,3,0} & 2\mu_{1,2,1} \\
\mu_{0,2,1} & \mu_{1,2,1} & \mu_{0,3,1} & \mu_{0,2,0} & \mu_{1,2,0} & \mu_{0,3,0} & 2\mu_{0,2,1} & \mu_{2,2,0} & \mu_{1,3,0} & 2\mu_{1,2,1} & \mu_{0,4,0} & 2\mu_{0,3,1} & 3\mu_{0,2,2} \\
\mu_{1,0,2} & \mu_{2,0,2} & \mu_{1,1,2} & 2\mu_{1,0,1} & 2\mu_{2,0,1} & 2\mu_{1,1,1} & 3\mu_{1,0,2} & 2\mu_{3,0,1} & 2\mu_{2,1,1} & 3\mu_{2,0,2} & 2\mu_{1,2,1} & 3\mu_{1,1,2} & 4\mu_{1,0,3} \\
\mu_{0,1,2} & \mu_{1,1,2} & \mu_{0,2,2} & 2\mu_{0,1,1} & 2\mu_{1,1,1} & 2\mu_{0,2,1} & 3\mu_{0,1,2} & 2\mu_{2,1,1} & 2\mu_{1,2,1} & 3\mu_{1,1,2} & 2\mu_{0,3,1} & 3\mu_{0,2,2} & 4\mu_{0,1,3} \\
\mu_{1,1,1} & \mu_{2,1,1} & \mu_{1,2,1} & \mu_{1,1,0} & \mu_{2,1,0} & \mu_{1,2,0} & 2\mu_{1,1,1} & \mu_{3,1,0} & \mu_{2,2,0} & 2\mu_{2,1,1} & \mu_{1,3,0} & 2\mu_{1,2,1} & 3\mu_{1,1,2}
\end{bmatrix}, b = -\begin{bmatrix}
\mu_{0,0,1} \\ \mu_{1,0,1} \\ \mu_{0,1,1} \\ \mu_{0,0,2} \\ \mu_{2,0,1} \\ \mu_{0,2,1} \\ \mu_{0,0,3} \\ \mu_{1,1,1} \\ \mu_{1,0,2} \\ \mu_{0,1,2} \\ \mu_{3,0,1} \\ \mu_{0,3,1} \\ \mu_{0,0,4} \\ \mu_{2,1,1} \\ \mu_{2,0,2} \\ \mu_{1,2,1} \\ \mu_{0,2,2} \\ \mu_{1,0,3} \\ \mu_{0,1,3} \\ \mu_{1,1,2}
\end{bmatrix}$$

and $x^T = \begin{bmatrix} a_0 & a_1 & a_2 & b_0 & b_1 & b_2 & b_3 & b_{11} & b_{12} & b_{13} & b_{22} & b_{23} & b_{33} \end{bmatrix}$. (5) becomes

$$a_0^\star = a_0 + a_1 Z_1 + a_2 Z_2, \quad b_0^\star = -(b_0 + b_1 Z_1 + b_{11} Z_1^2 + b_2 Z_2 + b_{22} Z_2^2 + b_{12} Z_1 Z_2)$$
$$b_1^\star = -(b_3 + b_{13} Z_1 + b_{23} Z_2), \quad b_{11}^\star = -b_{33}.$$

Finally, conditional moments can be computed according to

$$\begin{aligned}
\mathrm{E}[Z_3|Z_2, Z_1] &= (b_1^\star - a_0^\star)(1 - 2b_{11}^\star)^{-1}, \\
\mathrm{E}[Z_3^2|Z_2, Z_1] &= ((2b_1^\star - a_0^\star)\mathrm{E}[Z_3|Z_2, Z_1] + b_0^\star)(1 - 3b_{11}^\star)^{-1}, \\
\mathrm{E}[Z_3^3|Z_2, Z_1] &= ((3b_1^\star - a_0^\star)\mathrm{E}[Z_3^2|Z_2, Z_1] + 2b_0^\star \mathrm{E}[Z_3|Z_2, Z_1])(1 - 4b_{11}^\star)^{-1}, \\
\mathrm{E}[Z_3^4|Z_2, Z_1] &= ((4b_1^\star - a_0^\star)\mathrm{E}[Z_3^3|Z_2, Z_1] + 3b_0^\star \mathrm{E}[Z_3^2|Z_2, Z_1])(1 - 5b_{11}^\star)^{-1}.
\end{aligned}$$

Given conditional moments, it is possible to draw a random sample from $(Z_3|Z_2, Z_1)$ through the Pearson's system of distributions.

EC.4. Motivating the simulation approaches proposed in Section 3 and their accuracy

Due to the generality of Algorithm 1, the approach we propose in Section 3 for the simulation of the various option pricing models is not the only possible. Alternative schemes could be developed for the same models. For example, in the Heston, Jacobi, and mean-reverting Heston, to obtain a sample from (X_T, V_T), we suggest to simulate first the terminal variance, V_T, then the integrated variance $\left(\int_0^T V_s ds | V_T\right)$ and finally the log-returns $\left(X_T | \int_0^T V_s ds, V_T\right)$. For Hull-White models, we suggest sampling the volatility Y_T first, then the integrated volatility $\left(\int_0^T Y_s ds | Y_T\right)$ and finally the log-returns $\left(X_T | \int_0^T Y_s ds, Y_T\right)$. One may wonder why the intermediate step is suggested. Indeed, in the Heston, Jacobi and mean-reverting Heston models, one could simply simulate the variance and then the log-returns conditionally on the terminal variance, avoiding the simulation of the integrated

variance. We suggest to implement the intermediate step to improve the accuracy of the methodology, as we discuss next. Alternatively, one may wonder why not introducing additional steps. For example, in the Jacobi model, log-returns are conditionally Gaussian given V_T, $\int_0^T V_t dt$, and $\int_0^T V_t^2 dt$. So, why not modifying Algorithm 3 to simulate also $\int_0^T V_t^2 dt$ and then sample the log-returns using normal random numbers generators? As part of this research, we have tested various approaches. We found that Algorithms 2–7 are the best for the following reason. Since log-returns are simulated via moment-matching, accuracy will be high only if the distribution information content is represented by few first moments. Otherwise, it may result inaccurate. This can be understood considering the Heston model, where $\left(X_T | \int_0^T V_s ds, V_T\right)$ is normally distributed, meaning that the first two moments are sufficient to characterize the distribution. Note that this is not the case of $(X_T|V_T)$, where more moments are needed. Inspired by Fusai and Tagliani (2002), we can assess this fact numerically since the distribution information content can be measured via the entropy of a distribution (or differential entropy since we consider continuous distributions). Given a sequence of moments $\{\mu_j\}_{j=0}^n$, the probability density function of the maximum entropy distribution is

$$f_{ME}^{(n)}(x) = \exp\left(\sum_{j=0}^{n} \lambda_j x^j\right)$$

where $\underline{\lambda} \in \mathbb{R}^{n+1}$ is the vector of Lagrange multipliers found by solving numerically the following convex optimization problem

$$\max_{\lambda} \sum_{j=0}^{n} \lambda_j \mu_j - \int_{-\infty}^{\infty} \exp\left(\sum_{j=0}^{n} \lambda_j x^j\right) dx.$$

Equipped with the probability density function, $f_{ME}^{(n)}$, it is possible to compute the entropy of the distribution: $H_n(X) = -\int_{-\infty}^{\infty} f_{ME}^{(n)}(x) \log(f_{ME}^{(n)}(x)) dx$. When adding more and more moments, the entropies satisfy the following inequalities

$$H_1(x) \geq H_2(x) \geq ... \geq H_n(x).$$

The sequence of differences $H_1(x) - H_2(x)$, $H_2(x) - H_3(x)$, ..., $H_{n-1}(x) - H_n(x)$, decays to zero. If it decays fast then the distribution information content is represented by few first moments, so that the approximations matching the given first moments fits well the distribution to be approximated. If it decays slowly then higher order moments bring important information and moment matching approaches based on the first few moments may be ineffective. We can therefore evaluate the information content numerically. We simulate the terminal variance 10^4 times (or terminal volatility, depending on the model). Conditionally on the random realization, we compute $H_n(X_T|V_T)$ for $n = 2, 3, 4$ and take the average across simulations. Then, we simulate also $\left(\int_0^T V_s ds | V_T\right)$, compute $H_n\left(X_T | \int_0^T V_s ds, V_T\right)$ and take the average across simulations. Together with the average entropies, we also report the decay, i.e., $H_n(\cdot) - H_{n-1}(\cdot)$. We report the numerical results in Table 3. To save

Table 3: Average entropy of the maximum entropy distribution of the log-returns conditionally on terminal variance, V_T (in the Heston, Jacobi, mean-reverting Heston models) and on terminal volatility, Y_T (in the advanced and classic Hull-White models), for varying number of moments matched, n.

	H1				H2			
	$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$		$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$	
n	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay
2	-0.67904	–	-1.01674	–	-0.23260	–	-0.51977	–
3	-0.73021	0.05118	-1.01674	0.00000	-0.26642	0.03383	-0.51977	0.00000
4	-0.76126	0.03105	-1.01674	0.00000	-0.28115	0.01473	-0.51977	0.00000

	J1				J2			
	$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$		$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$	
n	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay
2	1.32359	–	-0.62737	–	1.19015	–	-0.17008	–
3	1.28695	0.03664	-0.62737	0.00000	1.17577	0.01438	-0.17008	0.00000
4	1.26639	0.02056	-0.62737	0.00000	1.17457	0.00121	-0.17008	0.00000

	J3				HW1			
	$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$		$(X_T\|Y_T)$		$(X_T\|\int_0^T Y_s ds, Y_T)$	
n	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay
2	1.35133	–	-0.19106	–	-0.33256	–	-0.29497	–
3	1.35024	0.00110	-0.19106	0.00000	-0.33398	0.00142	-0.29498	0.00000
4	1.34951	0.00072	-0.19106	0.00000	-0.33440	0.00042	-0.29498	0.00000

	HW2				HW3			
	$(X_T\|Y_T)$		$(X_T\|\int_0^T Y_s ds, Y_T)$		$(X_T\|Y_T)$		$(X_T\|\int_0^T Y_s ds, Y_T)$	
n	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay
2	0.11793	–	-0.07201	–	-1.23795	–	-1.26767	–
3	0.09741	0.02052	-0.07227	0.00000	-1.23816	0.00021	-1.26768	0.00000
4	0.09405	0.00336	-0.07233	0.00000	-1.23899	0.00083	-1.26768	0.00000

	MRH1				MRH2			
	$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$		$(X_T\|V_T)$		$(X_T\|\int_0^T V_s ds, V_T)$	
n	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay	Avg. Entropy	Decay
2	-0.79065	–	-1.11247	–	-0.34143	–	-0.61387	–
3	-0.84086	0.05021	-1.11278	0.00031	-0.37453	0.03311	-0.61390	0.00003
4	-0.87159	0.03073	-1.11288	0.00010	-0.38867	0.01414	-0.61391	0.00001

Notes. Parameters as in Table 1.

space, we omitted the Pilipović model (similar results are expected). Obviously, in the case of the Heston model, the first two moments completely characterize the distribution of $\left(X_T|\int_0^T V_s ds, V_T\right)$. Indeed, the entropy is unchanged when considering 2, 3 or 4 moments. This is not the case when considering $(X_T|V_T)$ where difference in entropy does not converge to 0 fast, meaning that more than four moments are necessary to obtain accurate results. Similar reasoning applies to the Jacobi and mean-reverting Heston models. For the advanced and classic Hull-White models, we find that the difference in entropies converges to 0 faster for $\left(X_T|\int_0^T Y_s ds, Y_T\right)$ than for $(X_T|Y_T)$. This means that it is easier to approximate $\left(X_T|\int_0^T Y_s ds, Y_T\right)$ via the first few integer moments. Also, the difference in entropies converges to 0 already for $n = 4$ (i.e., four moments matched) implying that the distribution information content is represented by four moments and explaining the high accuracy of our proposed simulation approach. Under the Jacobi model, note that decay is very fast, meaning that the first four integer moments are sufficient to characterize the distribution of $\left(X_T|V_T, \int_0^T V_s ds\right)$, eliminating the necessity, for example, to simulate also $\int_0^T V_s^2 ds$.

EC.5. Extensions and additional applications in financial engineering

Until now, we focused on the simulation of option pricing models and the pricing of European options (under stochastic volatility models) and Asian options (under the Pilipović model). The methodology can be also applied to alternative models. All affine stochastic volatility models (see, e.g., Hubalek *et al.* (2017) for a review) can be simulated via our approach since affine processes are special cases of polynomial processes. In this section, we devise some further applications of our general simulation approach. First, we show how to simulate models with jumps. Second, we present an example regarding the simulation of multi-factor models. Third, we consider the simulation of a model proposed in actuarial sciences. Fourth, we consider the simulation of the integrated Brennan-Schwartz process and discuss some applications. Fifth, we discuss the simulation of a certain class of non-polynomial models.

EC.5.1. Models with jumps

We start by considering a model with jumps in the price process. For simplicity, let us consider the **?** model (it is trivial to extend to the other models considered in the paper), that is given as follows

$$dX_t = \left(r - \frac{1}{2}V_t - \mu^\star \lambda_t\right)dt + \sqrt{V_t}\left(\rho dW_t^{(2)} + \sqrt{1-\rho^2}dW_t^{(1)}\right) + \mathcal{J}dN_t, \tag{3}$$

$$dV_t = k(\theta - V_t)dt + \sigma\sqrt{V_t}dW_t^{(2)}, \tag{4}$$

where $W_t^{(j)}$ are mutually independent standard Brownian motions, N_t is an independent Poisson process with jump intensity λ, the log-return jump size is $\mathcal{J} \sim \mathcal{N}(\mu_{\mathcal{J}}, \sigma_{\mathcal{J}}^2)$, and $\mu^\star = e^{\mu_{\mathcal{J}}+0.5\sigma_{\mathcal{J}}^2} - 1$ is the convexity adjustment for the jump component. Note that, if there are no jumps, conditionally on $\left(\int_0^T V_s ds, V_T\right)$, the log-returns are normally distributed:

$$\tilde{X}_T \sim \mathcal{N}(m, s^2), \tag{5}$$

where $m = rT - \lambda\mu^\star - \frac{1}{2}\int_0^T V_s ds + \frac{\rho}{\sigma}\left(V_T - V_0 - k\theta T + k\int_0^T V_s ds\right)$ and $s^2 = (1-\rho^2)\int_0^T V_s ds$. Thus, the model (3)–(4) can be simulated via a simple modification of Algorithm 2. To sample the log-returns at time T, the jump and the diffusion components can be simulated separately: we can first simulate the diffusion part (e.g., via Algorithm 2), and then add the realized jump sizes, i.e., $X_T = \tilde{X}_T + \sum_{j=1}^{N_t} \mathcal{J}_j$. This makes it possible to extend our simulation approach to the Bates model. The procedure is summarized into Algorithm 1.

We have implemented Algorithm 1 to simulate the Bates model using the same model parameters reported in Broadie and Kaya (2006, Table 6), i.e., $S_0 = 100$, $K = 100$, $V_0 = 0.008836$, $k = 3.99$, $\theta = 0.014$, $\sigma = 0.27$, $\rho = -0.79$, $\lambda = 0.11$, $\mu^\star = -0.12$, $\sigma_{\mathcal{J}} = 0.15$, $r = 3.19\%$, $T = 5$. Running 10^{10} simulations, we obtained an option price equal to 20.1635, whereas the true option price is 20.1642

Algorithm 1 Conditional moments-matched sampling scheme for the Bates model

Input: $\{r, V_0, k, \theta, \sigma, \rho, \lambda, \mu_{\mathcal{J}}, \sigma_{\mathcal{J}}\}$ (Model parameters), T (maturity), $\mathcal{M}$ (number of simulations)

Output: $\left\{\left(X_T^{(i)}, V_T^{(i)}\right)\right\}_{i=1}^{\mathcal{M}}$ (samples from variance and log-returns)

Compute $\mathrm{E}\left[V_T^{r_1}\left(\int_0^T V_s ds\right)^{r_2}\right]$ for $r_1, r_2 = 0, ..., 4$ such that $r_1 + r_2 \leq 4$
Solve (4) for $d = 2$
for $i = 1 : \mathcal{M}$ **do**
    Simulate $V_T^{(i)}$ from (7)
    Compute $\mathrm{E}\left[\left(\int_0^T V_s ds\right)^n | V_T\right]$ for $n = 1, ..., 4$ from (6)
    Simulate $\left(\int_0^T V_s ds\right)^{(i)}$ conditionally on V_T from moments-matched Pearson
    Simulate $\tilde{X}_T^{(i)}$ given $\left(\int_0^T V_s ds\right)^{(i)}$ and $V_T^{(i)}$ from (5)
    Simulate Generate a Poisson random variable with mean λT and call this number N_T
    Generate N_T independent jump sizes $\mathcal{J}_j$ for $j = 1, ..., N_T$, from the normal distribution with $\mu_{\mathcal{J}}$ and variance $\sigma_{\mathcal{J}}^2$
    Find the adjusted final log-return $X_T^{(i)} = \tilde{X}_T^{(i)} + \sum_{j=1}^{N_T} \mathcal{J}_j$
end

(see Broadie and Kaya, 2006, Table 6). This means that the bias in absolute value is 0.0007. To have a comparison, Kyriakou *et al.* (2024) obtain an absolute bias equal to 0.0005 for their methodology using the same parametrization. Therefore, when comparing Algorithm 1 against the methodology of Kyriakou *et al.* (2024) we obtain speed-accuracy profiles very similar to those reported for the Heston model in Section 4.2 (omitted to save space).

One could also add jumps in the variance process. In this case, the proposed methodology can be also applied. We just need to take into account jumps when computing the moments of the various distributions involved. We refer to Filipović *et al.* (2013) for an example.

EC.5.2. Multi-factor models

Our methodology can be also used to simulate efficiently multifactor models. For sake of illustration, following Recchioni *et al.* (2021), let us consider the Multi-Heston model:

$$dX_t = \left(r - \frac{1}{2}\sum_{j=1}^{J} V_t^{(j)}\right) dt + \sum_{j=1}^{J} \sqrt{V_t^{(j)}} \left(\rho_j dW_t^{(2j)} + \sqrt{1-\rho_j^2} dW_t^{(j)}\right) \tag{6}$$

$$dV_t^{(j)} = k_j(\theta_j - V_t^{(j)})dt + \sigma_j \sqrt{V_t} dW_t^{(2j)} \tag{7}$$

where $W_t^{(j)}$ are mutually independent standard Brownian motions. Let us define

$$\xi_1^{(j)} := \left(\frac{r}{J} - \frac{\rho_j k_j \theta_j}{\sigma_j}\right) T - \frac{\rho_j V_0^{(j)}}{\sigma_j}, \quad \xi_2^{(j)} := \frac{\rho_j}{\sigma_j}, \quad \xi_3^{(j)} := \frac{\rho_j k_j}{\sigma_j} - \frac{1}{2}, \quad \xi_4^{(j)} := (1-\rho_j^2).$$

The conditional distribution of the log–returns is

$$\left(X_T \middle| \left\{V_T^{(j)}, \int_0^T V_s^{(j)} ds\right\}_{j=1}^{J}\right) \sim \mathcal{N}(\underline{m}, \underline{s}^2) \tag{8}$$

with

$$\underline{m} = \sum_{j=1}^{J} \xi_1^{(j)} + \xi_2^{(j)} V_T^{(j)} + \xi_3^{(j)} \int_0^T V_s^{(j)} ds, \ \underline{s}^2 = \sum_{j=1}^{J} \xi_4^{(j)} \int_0^T V_s^{(j)} ds.$$

Let us consider the case $J = 2$ that corresponds to the so-called Double Heston model, introduced by Christoffersen *et al.* (2009). It can be simulated via a simple modification of Algorithm 2 as follows. We start by simulating $\left(\int_0^T V_s^{(1)} ds, V_T^{(1)}\right)$ and $\left(\int_0^T V_s^{(2)} ds, V_T^{(2)}\right)$ as outlined in Algorithm 2. Then, we sample the log-returns from (8). The procedure is summarized into Algorithm 2. Also in this case, the methodology turns out to be strictly related to the one proposed by Kyriakou *et al.* (2024). Numerical results are similar to those presented in the case of the Heston model (see Section 4.2) and are omitted to save space. We have considered the Double Heston model. However, we stress that it could be perfectly possible to simulate alternative multifactor models, such as for instance the multifactor Ornstein–Uhlenbeck driven stochastic volatility model, see Brignone (2024). This is omitted to save space.

Algorithm 2 Conditional moments-matched sampling scheme for the Double Heston model

Input: $\{r, V_0^{(1)}, k_1, \theta_1, \sigma_1, \rho_1, V_0^{(2)}, k_2, \theta_2, \sigma_2, \rho_2\}$ (Model parameters), T (maturity), $\mathcal{M}$ (number of simulations)

Output: $\left\{\left(X_T^{(i)}, V_T^{(1,i)}, V_T^{(2,i)}\right)\right\}_{i=1}^{\mathcal{M}}$ (samples from variance and log-returns)

Compute $\mathrm{E}\left[(V_T^{(1)})^{r_1}\left(\int_0^T V_s^{(1)} ds\right)^{r_2}\right]$ for $r_1, r_2 = 0, ..., 4$ such that $r_1 + r_2 \leq 4$
Solve (4) for $d = 2$
Compute $\mathrm{E}\left[(V_T^{(2)})^{r_1}\left(\int_0^T V_s^{(2)} ds\right)^{r_2}\right]$ for $r_1, r_2 = 0, ..., 4$ such that $r_1 + r_2 \leq 4$
Solve (4) for $d = 2$
for $i = 1 : \mathcal{M}$ **do**
    Simulate $V_T^{(1,i)}$ from (7)
    Simulate $V_T^{(2,i)}$ from (7)
    Compute $\mathrm{E}\left[\left(\int_0^T V_s^1 ds\right)^n | V_T^{(1)}\right]$ for $n = 1, ..., 4$ from (6)
    Compute $\mathrm{E}\left[\left(\int_0^T V_s^2 ds\right)^n | V_T^{(2)}\right]$ for $n = 1, ..., 4$ from (6)
    Simulate $\left(\int_0^T V_s^{(1)} ds\right)^{(i)}$ conditionally on $V_T^{(1)}$ from moments-matched Pearson
    Simulate $\left(\int_0^T V_s^{(2)} ds\right)^{(i)}$ conditionally on $V_T^{(2)}$ from moments-matched Pearson
    Simulate $X_T^{(i)}$ from (8)
end

EC.5.3. Simulation of the Biagini and Zhang model

Biagini and Zhang (2016) propose using polynomial processes to model some key quantities in insurance markets. In particular, for the purpose of pricing and hedging life insurance liabilities they propose using polynomial processes to accurately describe the time evolution of a benchmark portfolio and a longevity index. At the core (we refer to Biagini and Zhang, 2016 for more details), the simulation of their model depends on the simulation of two key quantities (X_T, Y_T) whose dynamics

are given as follows (cfr. Biagini and Zhang, 2016, Section 5):

$$dX_t = \Psi(b - X_t)dt + \sigma\sqrt{1 - X_t^2}dB_t$$
$$dY_t = (\tilde{d}(b - X_t) + k(\eta - Y_t))dt + \sigma\sqrt{1 - X_t^2}dB_t.$$

with $\sigma > 0$, Ψ, b, $\tilde{d}$, k, $\eta \in \mathbb{R}$ and the parameters satisfy the following condition $b\Psi x - \Psi x^2 \leq 0$, for $x \in (1, -1)$. X_t is an example of polynomial diffusions on unit ball, see also Larsson and Pulido (2017). Since (X_T, Y_T) enjoys the polynomial property, the model can be simulated according to the methodology proposed in this paper. We recover the expression in (1) by setting

$$Z_t = \begin{bmatrix} X_t \\ Y_t \end{bmatrix}, \quad \mu = \begin{bmatrix} \Psi(b - X_t) \\ \tilde{d}(b - X_t) + k(\eta - Y_t) \end{bmatrix}, \quad \underline{\sigma} = \begin{bmatrix} \sigma\sqrt{1 - X_t^2} \\ \sigma\sqrt{1 - X_t^2} \end{bmatrix}, \quad B_t = W_t.$$

Therefore, the model can be simulated as a special case of Algorithm 1. First, we compute the joint moments of (X_T, Y_T) up to order four. Second, we simulate X_T from a four moments matched Pearson. Third, we set up and solve the linear system in (4) for $d = 2$ and compute the first four integer moments of $(Y_T|X_T)$. Finally, we simulate $(Y_T|X_T)$ from a four moments matched Pearson distribution.

Next, we test our proposed simulation scheme for the Biagini and Zhang model. We take the same model parameters calibrated on real data by Biagini and Zhang (2016, p. 125): $\Psi = 15$, $b = -0.8$, $\sigma = 1.25$, $d = 5.2$, $k = -5.9$, $\eta = -5$, $X_0 = Y_0 = 0$. We fix $T = 1$ and simulate 10^6 times the couple (X_T, Y_T) using our proposed approach and the Euler scheme (used as benchmark). Given random samples from both simulation methods, we implement a two-sample Kolmogoroff-Smirnov test where the null hypothesis is that samples come from the same continuous distribution. Results are reported in Table 4 for varying number of time discretization steps, n, in the Euler scheme. In this table we report the p-value, and the maximum distance in absolute value $\bar{k}$. We notice that for small number of time steps (e.g., $n = 100$) we strongly reject the null hypothesis (both for X_T and Y_T). However, increasing n we observe that the p-value increases and $\bar{k}$ decreases meaning that the cumulative distribution functions get closer and closer. Finally, for $n = 3200$ we fail to reject the null hypothesis for Y_T, meaning that the sample obtained via our proposed method is highly accurate and statistically indistinguishable from the one obtained via Euler scheme with $n = 3200$ time steps. However, the proposed approach is extremely faster, requiring only 1.28 seconds to perform 10^6 simulations, while the Euler scheme runs in 25.15 seconds for the same number of simulations with $n = 3200$ time steps.

EC.5.4. Simulation of the integrated generalized Brennan-Schwartz process and applications

Consider the following model (hereafter, generalized Brennan-Schwartz process)

$$dX_t = k(\theta - X_t)dt + (\nu + \sigma X_t)\, dW_t. \tag{9}$$

Table 4: Simulation of the Biagini and Zhang model: results of the two sample Kolmogoroff-Smirnov test.

	X_T			Y_T		
n	p-value	$\bar{k}$	h	p-value	$\bar{k}$	h
100	9.95E-21	0.0216	1	4.37E-148	0.0184	1
200	2.62E-04	0.0095	1	1.11E-29	0.0082	1
400	0.1860	0.0049	0	5.21E-10	0.0047	1
800	0.1934	0.0048	0	7.44E-05	0.003193	1
1600	0.5868	0.0035	0	0.025	0.0021	1
3200	0.8075	0.0029	0	0.746	0.00096	0

Notes. n denotes the number of time steps for implementing the Euler scheme. $\bar{k}$ denotes the test statistic, h is the hypothesis test result: if $h = 1$, this indicates the rejection of the null hypothesis that the data in vectors are from the same continuous distribution at the 5% significance level; if $h = 0$ this indicates a failure to reject the null hypothesis at the 5% significance level.

When $\nu = 0$ the process X_t follows the Brennan and Schwartz (1980) process, also GARCH diffusion process. This process is widely used in finance. For example, in the case $\nu = 0$, it is used to model interest rate uncertainty (see Kyriakou *et al.*, 2024), to model default intensity in credit risk (Li *et al.* (2018)), to model mean-reverting asset prices and price Asian options (Cai *et al.* (2014)). In the general case $\nu \neq 0$, Ackerer and Filipović (2020) use (9) to model the variance of asset returns. Kyriakou *et al.* (2024) propose a conditional moments-matched sampling scheme for $(X_T|X_0)$ which works only in the case with $\nu = 0$. Our methodology is much more general since it can be applied to simulate the couple $\left(X_T, \int_0^T X_s ds\right)$ in the case where $\nu \neq 0$. Defining $Y_t := \int_0^t X_s ds$, we recover the expression in (1) by setting

$$Z_t = \begin{bmatrix} X_t \\ Y_t \end{bmatrix}, \quad \mu = \begin{bmatrix} k(\theta - X_t) \\ X_t \end{bmatrix}, \quad \underline{\sigma} = \begin{bmatrix} \nu + \sigma X_t \\ 0 \end{bmatrix}, \quad B_t = W_t.$$

The model can be simulated as a special case of Algorithm 1. We compute the joint moments of (X_T, Y_T) up to order four. Then, we simulate X_T from a four moments matched Pearson, set up and solve the linear system in (4) for $d = 2$ and compute the first four integer moments of $(Y_T|X_T)$. Finally, we simulate $(Y_T|X_T)$ from a four moments matched Pearson distribution. This can be applied in various contexts. For example, computing the probability of default in credit risk (see Li *et al.* (2018)), the pricing of Asian options (similarly to Cai *et al.* (2014)), and the pricing of options on variance (as in Ackerer and Filipović (2020)). For illustration, let us consider the problem of pricing options on variance in the GARCH model (as in Ackerer and Filipović (2020)). We take the same model parameters used in Ackerer and Filipović (2020): $T = 1/2$, $k = 0.5$, $\theta = 0.2$, $X_0 = 0.2$, $\nu = -0.025$, $\sigma = 0.5$, $r = 0$ and a strike price $\bar{k} = 0.2$. The price of the option is computed as

$$C_{Var} = e^{-rT} \mathrm{E}\left[\max\left(0, \frac{\int_0^T V_s ds}{T} - \bar{k}\right)\right] \times 10^2.$$

To compute the price we simulate 10^7 times the couple (X_T, Y_T) with our approach and via the Euler scheme (with 3200 time steps). We obtain a price equal to (standard error in parenthesis): 1.1099 (5.99×10^{-4}) with our proposed approach, 1.1105 (5.94×10^{-4}) with the Euler scheme. Therefore,

the proposed approach gives accurate results (compare also with (Ackerer and Filipović, 2020, Figure 3)). Additionally, it is much faster than the Euler scheme. Indeed, for 10^7 simulations it takes 11.46 seconds, while the benchmark Euler scheme takes 950.04 seconds.

EC.5.5. Extending to non-polynomial models

Zhao *et al.* (2025) show that polynomial models can serve as approximations for more general dynamics. More specifically, Zhao *et al.* (2025) consider the following k-th order approximation

$$\mathrm{E}\left[e^{-\int_0^T r(X_s)ds}\langle \bar{f}^k, b^k(X_t)\rangle | X_0 = x\right] \approx \langle e^{TA_k}, b^k(x)\rangle, \tag{10}$$

where $b^k(x)$ is a vector of basis functions evaluated at x, $\bar{f}^k$ is a vector of length k and $\langle \bar{f}^k, b^k(X_t)\rangle = \sum_{i=0}^k f_i b_i(x)$. The matrix A_k can be derived from the infinitesimal generator of the process X_t and the function $r(\cdot)$. When (X_t) is a polynomial process, f is a polynomial function, and $r = 0$ (no discounting), then (10) holds with equality. This is the case considered in this paper. However, it is possible to use (10) to approximate the moments of $(X_T|X_0)$ also for some non-polynomial processes. The polynomial approximation theory of Zhao *et al.* (2025) works for a kind of Feller processes called *sequential* processes, which is more general than polynomial processes. Indeed, from (Zhao *et al.*, 2025, Definition 3.1 and Remark 3.2) we have that all polynomial processes are sequential (we refer to the original paper for more details). Therefore, consider a multivariate sequential process (X_t). It is possible (at least in principle) to approximate the joint moments using (10) and then use Algorithm 1 to obtain approximate samples from $(X_T|X_0)$. As future research, we plan to exploit the results of Zhao *et al.* (2025) to extend the range of models under which the proposed methodology is applicable by designing new applications in financial engineering.

EC.6. Illustrative example in the Heston model

In this section, we provide a detailed step by step guide to implement the proposed approach in the Heston model (i.e., Algorithm 2). We also compare the results with the ones obtained implementing the methodology of Kyriakou *et al.* (2024). To implement our simulation scheme, we first need to compute the joint moments of $\mathrm{E}\left[V_T^{r_1}\left(\int_0^T V_s ds\right)^{r_2}\right] =: \mu_{r_1,r_2}$. We obtain them running the Matlab® code provided in Section EC.2. Given the matrix G computed in this way, it is possible to compute μ_{r_1,r_2} by taking the exponential of a matrix, e^{GT}. Given joint moments, we standardize them applying the multinomial theorem:

$$\tilde{\mu}_{r_1,r_2} = \frac{1}{\left(\sqrt{\mu_{2,0}-\mu_{1,0}^2}\right)^{r_1}\left(\sqrt{\mu_{0,2}-\mu_{0,1}^2}\right)^{r_2}} \sum_{k_1=0}^{r_1}\sum_{k_2=0}^{r_2}\binom{r_1}{k_1}\binom{r_2}{k_2}\mu_{k_1,k_2}(-\mu_{1,0})^{r_1-k_1}(-\mu_{0,1})^{r_2-k_2}.$$

The values of μ_{r_1,r_2} and $\tilde{\mu}_{r_1,r_2}$ for the parameter set H1 given in Table 1 are reported in Table 5. Given standardized moments, we recover the matrix A and the vector b for the system (4) according to (compare with Section 2.3):

Table 5: Joint and standardized joint moments of $\left(V_T, \int_0^T V_s ds\right)$ in the Heston model.

μ_{r_1,r_2}	$\tilde{\mu}_{r_1,r_2}$	r_1	r_2
0.018982321	0	1	0
0.000928504	1	2	0
7.32E-05	2.511432291	3	0
7.96E-06	12.46092724	4	0
0.017585939	0	0	1
0.000424229	0.338112964	1	1
2.52E-05	0.84913934	2	1
2.33E-06	4.213137254	3	1
0.0004351	1	0	2
1.31E-05	0.556913267	1	2
9.21E-07	2.987721058	2	2
1.48E-05	1.908441361	0	3
5.42E-07	2.661418807	1	3
6.61E-07	8.97734262	0	4

Notes. Parameters as in H1 (see Table 1).

$$A = \begin{bmatrix} 1.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.676 \\ 0.000 & 0.338 & 1.000 & 0.000 & 0.000 & 1.000 & 0.676 & 3.000 \\ 1.000 & 2.511 & 0.000 & 0.000 & 1.000 & 0.000 & 2.511 & 1.698 \\ 0.338 & 0.849 & 0.000 & 1.000 & 0.676 & 2.511 & 1.698 & 1.671 \\ 1.000 & 0.557 & 0.000 & 0.676 & 3.000 & 1.698 & 1.671 & 7.634 \\ 2.511 & 12.461 & 0.000 & 0.000 & 2.511 & 0.000 & 12.461 & 8.426 \\ 0.849 & 4.213 & 1.000 & 2.511 & 1.698 & 12.461 & 8.426 & 8.963 \\ 0.557 & 2.988 & 0.676 & 1.698 & 1.671 & 8.426 & 8.963 & 10.646 \\ 1.908 & 2.661 & 3.000 & 1.671 & 7.634 & 8.963 & 10.646 & 44.887 \end{bmatrix}, \quad b = - \begin{bmatrix} 0.000 \\ -0.338 \\ -1.000 \\ -0.849 \\ -0.557 \\ -1.908 \\ -4.213 \\ -2.988 \\ -2.661 \\ -8.977 \end{bmatrix}.$$

We have rounded to the third digit to save space. We solve the system $Ax = b$ numerically obtaining

$$x^T = \begin{bmatrix} 0.7464 & -0.3011 & -0.8012 & 0.0083 & -0.7464 & 0.0089 & -0.0155 & -0.0318 \end{bmatrix}.$$

Given x, we can recover the coefficients as follows: $a_0 = 0.7464$, $a_1 = -0.3011$, $b_0 = -0.8012$, $b_1 = 0.0083$, $b_2 = -0.7464$, $b_{11} = 0.0089$, $b_{12} = -0.0155$, $b_{22} = -0.0318$.

We can now start our simulation and compare with the methodology in Kyriakou *et al.* (2024). For illustration, we draw 1 random sample for the terminal variance from (7) (this step is unchanged between the two methodologies). We obtain (rounding at the fourth decimal place) $V_T = 0.0034$. Then, to implement our method we standardize the variance: $\tilde{V}_T = \frac{V_T - \mu_{1,0}}{\sqrt{\mu_{2,0} - \mu_{1,0}^2}} = -0.6530$. Next, we

compute

$$
\begin{aligned}
a_0^\star &= a_0 + a_1\tilde{V} = 0.7464 + 0.3011 \times 0.6530 = 0.9430 \\
b_0^\star &= -(-0.8012 - 0.0083 \times 0.6530 + 0.0089 \times 0.6530^2) = 0.8028, \\
b_1^\star &= -(b_2 + b_{12}\tilde{V}) = -(-0.7464 - 0.0155 \times (-0.6530)) = 0.7363, \quad b_{11}^\star = 0.0318.
\end{aligned}
$$

Given such values, we can compute the standardized moments. Define the standardized integrated variance as $\tilde{L} := \frac{\int_0^T V_s ds - \mu_{0,1}}{\sqrt{\mu_{0,2} - \mu_{0,1}^2}}$, the conditional moments of $\tilde{L}$ are given as follows

$$
\begin{aligned}
\mathrm{E}[\tilde{L}|\tilde{V}] &= (b_1^\star - a_0^\star)(1 - 2b_{11}^\star)^{-1} = -0.2208, \\
\mathrm{E}[\tilde{L}^2|\tilde{V}] &= ((2b_1^\star - a_0^\star)\mathrm{E}[\tilde{L}|\tilde{V}] + b_0^\star)(1 - 3b_{11}^\star)^{-1} = 0.7582, \\
\mathrm{E}[\tilde{L}^3|\tilde{V}] &= ((3b_1^\star - a_0^\star)\mathrm{E}[\tilde{L}^2|\tilde{V}] + 2b_0^\star\mathrm{E}[\tilde{L}|\tilde{V}])(1 - 4b_{11}^\star)^{-1} = 0.6934, \\
\mathrm{E}[\tilde{L}^4|\tilde{V}] &= ((4b_1^\star - a_0^\star)\mathrm{E}[\tilde{L}^3|\tilde{V}] + 3b_0^\star\mathrm{E}[\tilde{L}^2|\tilde{V}])(1 - 5b_{11}^\star)^{-1} = 3.8221.
\end{aligned}
$$

At this point, defining $s := \sqrt{\mu_{0,2} - \mu_{0,1}^2} = \sqrt{0.0004351 - 0.017585939^2} = 0.0112$, we de-standardize the moments obtaining:

$$
\begin{aligned}
\mathrm{E}\left[\int_0^T V_s ds \middle| V_T\right] &= \mu_{0,1} + s\mathrm{E}[\tilde{L}|\tilde{V}] = 0.0176 - 0.0112 \times 0.2208 = 0.0151, \\
\mathrm{E}\left[\left(\int_0^T V_s ds\right)^2 \middle| V_T\right] &= s^2\mathrm{E}[\tilde{L}^2|\tilde{V}] + 2\mu_{0,1}\mathrm{E}\left[\int_0^T V_s ds \middle| V_T\right] - \mu_{0,1}^2 = 0.0003, \\
\mathrm{E}\left[\left(\int_0^T V_s ds\right)^3 \middle| V_T\right] &= s^3\mathrm{E}[\tilde{L}^3|\tilde{V}] + 3\mu_{0,1}\mathrm{E}\left[\left(\int_0^T V_s ds\right)^2 \middle| V_T\right] - 3\mu_{0,1}^2\mathrm{E}\left[\int_0^T V_s ds \middle| V_T\right] + \mu_{0,1}^3 \\
&= 9.15 \times 10^{-6}, \\
\mathrm{E}\left[\left(\int_0^T V_s ds\right)^4 \middle| V_T\right] &= s^4\mathrm{E}[\tilde{L}^4|\tilde{V}] + 4\mu_{0,1}\mathrm{E}\left[\left(\int_0^T V_s ds\right)^3 \middle| V_T\right] - 6\mu_{0,1}^2\mathrm{E}\left[\left(\int_0^T V_s ds\right)^2 \middle| V_T\right] + \\
&\quad 4\mu_{0,1}^3\mathrm{E}\left[\int_0^T V_s ds \middle| V_T\right] - \mu_{0,1}^4 = 3.48 \times 10^{-7}.
\end{aligned}
$$

For the same random realization of the variance, we can compare the moments computed in this way with those obtained via the method of Kyriakou *et al.* (2024) (see below). We can now perform moment-matching with the Pearson distribution. We use the built-in Matlab function `pearsrnd` for family selection and random sampling. The simulated value of the conditional integrated variance with our method is 0.0224 against 0.0225 with the method of Kyriakou *et al.* (2024) (we set the same seed for random sampling). Finally, given the sample from $\left(V_T, \int_0^T V_s ds\right)$, we sample $\left(X_T|V_T, \int_0^T V_s ds\right)$ from (8). The obtained sample is 0.0760 via our method and 0.0755 via the method of Kyriakou *et al.* (2024). We repeat this experiment 10 times, results are reported in Table 6. Finally, we compare the moments of $\left(\int_0^T V_s ds|V_T\right)$ computed via both methods in Table 7 and show that the proposed approach produces extremely accurate approximations for the conditional

Table 6: Illustrative example: 10 simulations of $\left(V_T, \int_0^T V_s ds, X_T\right)$ using Algorithm 2 and the method of Kyriakou *et al.* (2024).

Repetition	1	2	3	4	5	6	7	8	9	10
V_T	0.0034	0.0012	0.0014	0.0168	0.0071	0.0190	0.0143	0.0091	0.0384	0.0230
$\int_0^T V_s ds$ (KBF)	0.0225	0.0066	0.0214	0.0089	0.0077	0.0176	0.0063	0.0143	0.0156	0.0160
$\int_0^T V_s ds$ (Algo 2)	0.0224	0.0066	0.0214	0.0089	0.0077	0.0176	0.0062	0.0143	0.0156	0.0161
X_T (KBF)	0.0755	0.1525	-0.0956	0.1532	0.0403	0.0372	0.0623	0.0264	0.0551	-0.0170
X_T (Algo 2)	0.0760	0.1525	-0.0956	0.1531	0.0403	0.0377	0.0625	0.0265	0.0549	-0.0173
Abs. Diff	0.0005	1.36E-05	1.42E-05	9.41E-05	2.98E-05	0.0005	0.0002	1.87E-05	0.0002	0.0003

Notes. Abs. Diff. denotes the difference in absolute value between the sample for X_T generated via the two methods.

moments since the error (in absolute value) appears only at the 8th-10th decimal place. This finding remains unchanged if we increase the number of simulations (see Section 4.2).

EC.7. Exact simulation of the terminal variance

In this section we show how to simulate exactly the terminal variance under the Jacobi and Advanced Hull-White models.

EC.7.1. Jacobi model

We develop an accurate simulation scheme for the terminal variance, V_T, in the Jacobi stochastic volatility model. The idea is to exploit two basic facts: *i*) the moments of V_T can be computed accurately and fast thanks to the polynomial property; *ii*) V_T is defined on a compact support, $[v_{min}, v_{max}]$, implying that the moment problem is determined (see discussion below). We define $\mu_V(j) := \mathrm{E}[V_T^j]$ the j-th moments of V_T. Arbel *et al.* (2016) develop an efficient simulation scheme for distributions defined on the compact support $[0, 1]$. Following Provost (2005), we adapt their methodology to the present case. We define a continuous random variable X defined on the interval $[-1, 1]$. The probability density function can be expressed as

$$f_X(x) = \sum_{k=0}^{\infty} \lambda_k P_k(x),$$

where

$$\lambda_k = \frac{2k+1}{2} \sum_{i=0}^{\mathrm{Floor}[k/2]} (-1)^i 2^{-k} \frac{(2k-2i)!}{i!(k-i)!(k-2i)!} \mu_X(k-2i),$$

$P_k(x)$ are Legendre Polynomials:

$$P_k(x) = \sum_{i=0}^{\mathrm{Floor}[k/2]} (-1)^i 2^{-k} \frac{(2k-2i)!}{i!(k-i)!(k-2i)!)} x^{k-2i},$$

Table 7: Illustrative example: computing moments of $\left(\int_0^T V_s ds \middle| V_T\right)$ using formula (6) and Kyriakou *et al.* (2024, Algorithm 1) under the Heston model.

	First moment			Second moment		
Repetition	KBF	(6)	Abs.Diff	KBF	(6)	Abs.Diff
1	1.51E-02	1.51E-02	6.58E-08	3.18E-04	3.18E-04	2.62E-09
2	1.47E-02	1.47E-02	7.94E-08	3.04E-04	3.04E-04	3.00E-09
3	1.48E-02	1.48E-02	7.72E-08	3.05E-04	3.05E-04	2.95E-09
4	1.72E-02	1.72E-02	2.38E-10	4.05E-04	4.05E-04	2.86E-10
5	1.57E-02	1.57E-02	4.82E-08	3.41E-04	3.41E-04	2.00E-09
6	1.76E-02	1.76E-02	9.78E-09	4.21E-04	4.21E-04	1.03E-10
7	1.68E-02	1.68E-02	1.16E-08	3.89E-04	3.89E-04	7.26E-10
8	1.60E-02	1.60E-02	3.69E-08	3.54E-04	3.54E-04	1.65E-09
9	2.07E-02	2.07E-02	8.22E-08	5.67E-04	5.67E-04	3.57E-09
10	1.82E-02	1.82E-02	2.71E-08	4.49E-04	4.49E-04	8.31E-10

	Third moment			Fourth moment		
Repetition	KBF	(6)	Abs.Diff	KBF	(6)	Abs.Diff
1	9.12E-06	9.15E-06	3.47E-08	3.45E-07	3.48E-07	3.44561E-09
2	8.56E-06	8.60E-06	4.68E-08	3.18E-07	3.23E-07	4.83073E-09
3	8.63E-06	8.67E-06	4.53E-08	3.21E-07	3.26E-07	4.66066E-09
4	1.28E-05	1.28E-05	2.46E-08	5.27E-07	5.23E-07	3.76177E-09
5	1.01E-05	1.01E-05	1.65E-08	3.90E-07	3.91E-07	1.31241E-09
6	1.35E-05	1.34E-05	3.24E-08	5.61E-07	5.56E-07	4.75455E-09
7	1.21E-05	1.21E-05	1.52E-08	4.90E-07	4.87E-07	2.56914E-09
8	1.06E-05	1.06E-05	7.16E-09	4.16E-07	4.16E-07	1.89336E-10
9	2.03E-05	2.02E-05	7.74E-08	9.29E-07	9.18E-07	1.08308E-08
10	1.48E-05	1.47E-05	4.53E-08	6.28E-07	6.22E-07	6.44866E-09

Notes. Abs. Diff. denotes the difference in absolute value. The simulated values of V_T are the same as in Table 6. Parameters as in H1 (see Table 1).

and $\mu_X(i) = \int_{-1}^{1} x^i f_X(x)dx$. The moments of X can be defined in terms of the moments of V_T. Indeed, we first note that

$$X = \frac{2V_T - (v_{min} + v_{max})}{v_{max} - v_{min}} \in [-1, 1],$$

then,

$$\mu_X(j) = \frac{1}{(v_{max} - v_{min})^j} \sum_{k=0}^{j} \binom{j}{k} 2^k \mu_V(k)(-1)^{j-k}(v_{min} + v_{max})^{j-k}.$$

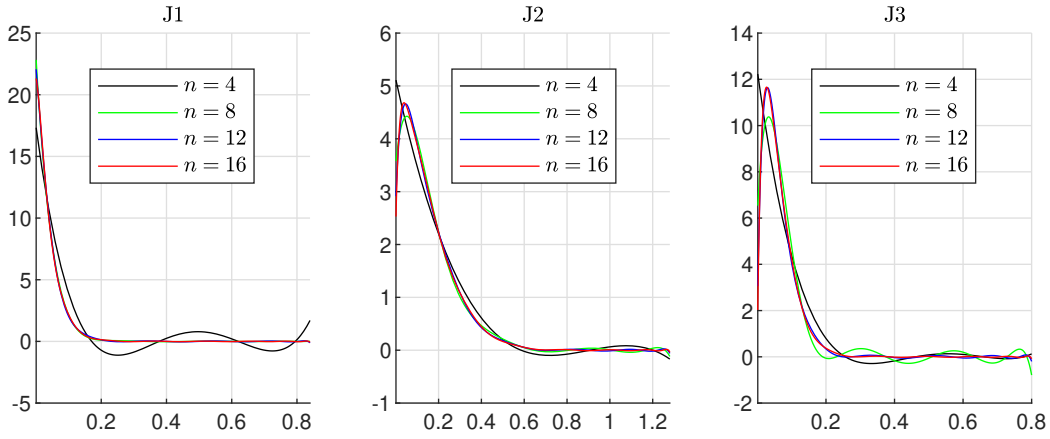
Finally, the density of V_T can be obtained according to

$$f_{V_T}(v) = \frac{2}{v_{max} - v_{min}} \sum_{k=0}^{\infty} \lambda_k P_k \left(\frac{2v - (v_{min} + v_{max})}{v_{max} - v_{min}} \right), \tag{11}$$

In practice, we need to truncate the infinite sum in (11). If we consider the first n integer moments, f_{V_T} can be approximated as

$$f_{V_T}(v) \approx f_{V_T}^{(n)}(v) = \frac{2}{v_{max} - v_{min}} \sum_{k=0}^{n} \lambda_k P_k \left(\frac{2v - (v_{min} + v_{max})}{v_{max} - v_{min}} \right). \tag{12}$$

Figure 1: $f_{V_T}^{(n)}$ for three different parameter sets and for varying n.



Notes. n denotes the number of moments matched.

A positive aspect of this approach is that, since V_T is defined on a compact support, it is theoretically guaranteed that the moment problem is determined. Therefore, $f_{V_T}(v) = \lim_{n\to\infty} f_{V_T}^{(n)}(v)$, ensuring that increasing the number of moments considered, the sequence of approximants converges to the true distribution. We remark that the density approximants can be negative in some regions of the domain (especially those with low density). This happens when an insufficient number of moments are being used. However, by inspection of the approximate density plot, one should be able to determine whether a higher degree polynomial should be used. Indeed, due to the convergence of the approximant, the density function will converge everywhere to a nonnegative number as more moments are used. We provide a graphical illustration in Figure 1.

Once we fix a number of moments n, we can then compute the approximate probability density function of f_{V_T}. At this stage, random sampling can be performed easily by accept/reject method (as done in Arbel *et al.* (2016)) or also by numerically integrating the density to obtain the cumulative distribution function and then applying inverse transform sampling. Figure 2 illustrates a simulation outcome with 16 moments matched and 10^5 samples generated. The algorithm runs in less than a second, which is still slower than applying moment-matching with Pearson. For Algorithm 3, we have used the latter approach. Indeed, the actual difference between the two approaches is very small, and we decided to stick to the Pearson based moment-matching for simplicity and to keep exposition as uniform as possible. To compare the samples obtained via the two approaches, we plot in Figure 3 the resulting cumulative distribution functions for the three parameter sets considered in Table 1. A two sample Kolmogoroff-Smirnov test indicates failure to reject the null hypothesis that the samples come from two different statistical distributions.

EC.7.2. Advanced Hull-White model

Under the model (12)–(13) with $\nu = 0$ the terminal volatility can be simulated exactly. Note, indeed, that under such model the volatility follows a linear SDE whose explicit solution is known from Kloeden and Platen (1992). This result has been exploited in (Kyriakou *et al.*, 2024, Section

Figure 2: Simulation of the terminal variance using accept-reject method

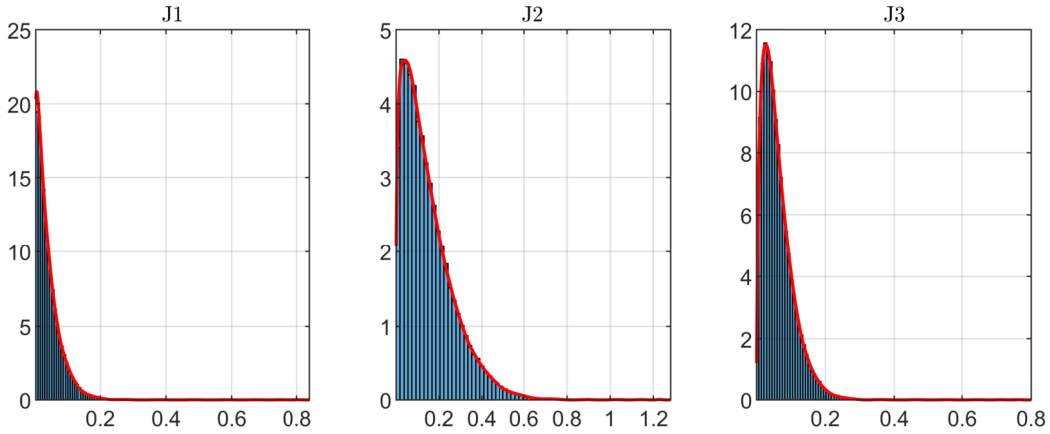
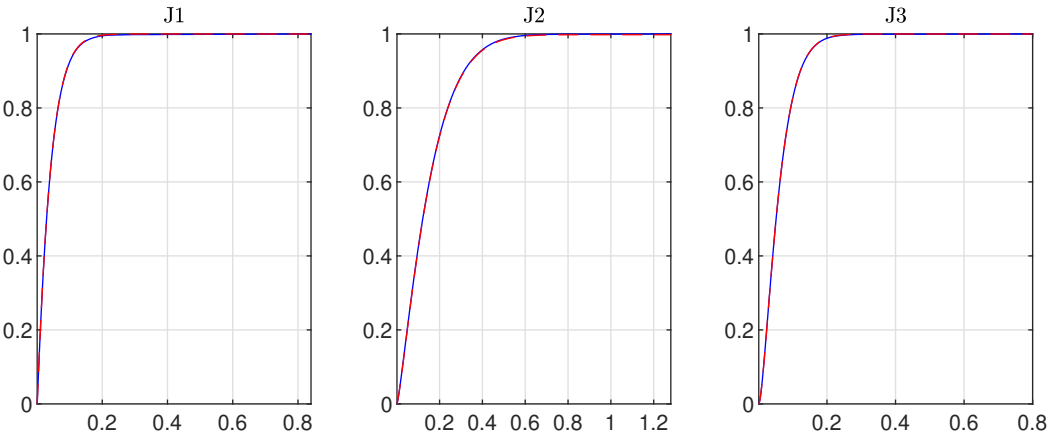


Figure 3: Distribution functions of the terminal variance, V_T, as simulated (almost) exactly from (12) with $n = 16$ and from the four moments matched distribution.



Notes. True distribution: blue continuous line; moment matched Pearson: red dashed lines.

6.1) to develop a conditional moments-matched sampling scheme. However, their methodology can be refined to obtain an exact sample. We rewrite (12) as

$$dY_t = (aY_t + c)dt + bY_t dW_t,$$

where $a = -k$, $b = \sigma$ and $c = \theta k$. From Kloeden and Platen (1992) we have

$$Y_T = \Upsilon\left(T; a, b, 1\right)\left(Y_0 + c\int_0^T \Upsilon\left(s; a, b-1\right) ds\right) \tag{13}$$

where $\Upsilon\left(T; a, b, \gamma\right) = \exp\left(\gamma\left(a - \frac{b^2}{2}\right)T + \gamma b W_T\right)$. From Matsumoto and Yor (2005) (see also Cai *et al.* (2017); Kyriakou *et al.* (2024)) we have

$$\mathrm{E}\left[\exp\left(-\frac{u}{\int_0^T \Upsilon(s;a,b,\gamma)ds}\right)\middle|\Upsilon\left(T;a,b,\gamma\right)\right] = \exp\left(-\frac{g\left(\frac{\log\Upsilon(T;a,b,\gamma)}{2}, \frac{\gamma^2 b^2 u}{4}\right)^2 - \left(\frac{\log(\Upsilon(T;a,b,\gamma)^2}{2}\right)^2}{\frac{\gamma^2 b^2 T}{2}}\right) \tag{14}$$

where $g(w,u) = \operatorname{arcosh}\left(ue^{-w} + \cosh(w)\right)$. Thus, Y_T can be simulated exactly as follows: *i*) simulate $\Upsilon(T,a,b,1)$; *ii*) Recover $\Upsilon(T,a,b,-1)$; *iii*) Simulate $\int_0^T \Upsilon(s;a,b,-1)ds$ given $\Upsilon(T,a,b,-1)$; *iv*) Recover Y_T from (13). At this stage, we just need to discuss how to simulate exactly

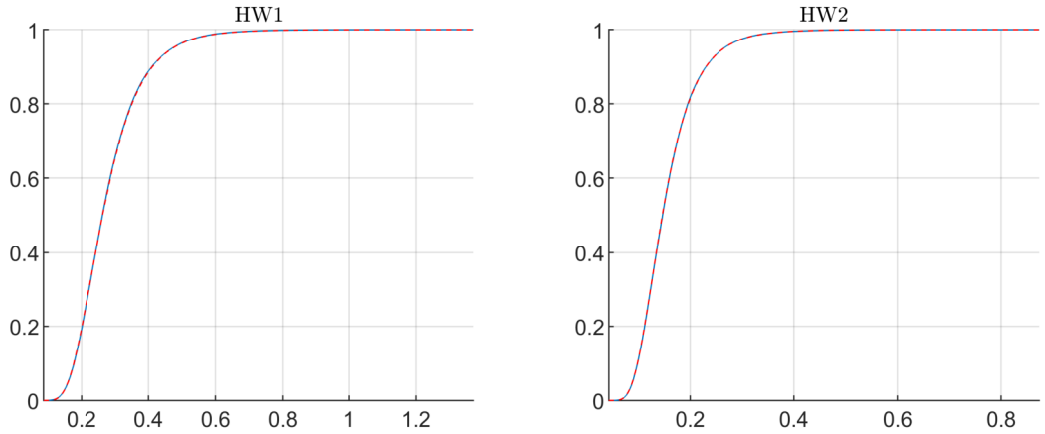
$$\left(\int_0^T \Upsilon(s;a,b,-1)ds|\Upsilon(T,a,b,-1)\right).$$

This is done by numerically inverting the Laplace transform in (14) to the cumulative distribution function and then perform random sampling using the inverse transform method. To this aim, we follow the same approach as in Brignone and Gonzato (2024) and use the COS method to obtain a numerical estimate of the cumulative distribution function. To save space, we do not provide a description of the methodology and rather refer to (Brignone and Gonzato, 2024, Section 3.2). In Figure 4 we report a comparison between the true cumulative distribution function of Y_T (obtained via exact simulation) and the one of the moments matched Pearson. The two distributions almost match. The two sample Kolmogoroff-Smirnov test indicates rejection of the null hypothesis that samples are from different distributions (p-value around 0.5 for both parameter sets considered). Thus, the Pearson's distribution provides a very good approximation to the true variance, as also clear from the results reported in Table 4.

EC.8. Additional results on the error bound

In this section we provide additional comments regarding the theoretical error bound in (24). First, it must be stressed that it is unclear if the error bound is finite, because the integral in (24) could be divergent. The complicated form of the integrand in our error bound prevents us from theoretically verifying that integral in (24) is convergent. Nevertheless, it is straightforward

Figure 4: Distribution functions of the terminal volatility, Y_T, as simulated exactly and from the four moments matched distribution.



Notes. True distribution: blue continuous line; moment matched Pearson: red dashed lines.

to numerically inspect this issue. We computed numerically (24) for various levels of $\delta_{F,\tilde{F}}$ and realistic values of μ_{2m} (that are unknown, except for the Heston model). In all the cases considered (unreported to save space), the built-in Matlab function `integral` provided reasonable values below the trivial bound $|P - \tilde{P}| < \min(S_0, Ke^{-rT})$. However, it should also be noticed that error bound is very conservative and not sharp. Indeed, it is well known in the literature that error bounds estimated obtained by studying proximity of distributions sharing the same number of moments are sharp only in the tails of the distribution, e.g. Lindsay and Basak (2000), leading generally to non-sharp bounds when applied e.g. to option pricing as in the present case. Moreover, in our context the theoretical analysis is further complicated by the fact that conditional moments are approximated and they do not match exactly. Under the parameter set H1, even assuming unrealistically that $\delta_{F,\tilde{F}} = 0$ we obtain an error bound estimate equal to 19.7893, whereas the true put option price is 3.6664. More realistically, setting $\delta_{F,\tilde{F}} = 10^{-9}$ (this value has been computed numerically comparing the true moments of the conditional log-returns with those estimated via the approach proposed in this paper), the error bound in (24) is 39.8348. Therefore, the provided bound can be interesting from a theoretical point of view, but difficult to use in practice. However, it is an improvement with respect to the trivial bound for a put option $|P - \tilde{P}| < \min(S_0, Ke^{-rT}) = 96.8603$, which is greater than 39.8348.

It should be noted that in the case of the Heston model, where the characteristic function of $(X_T|V_T)$ is known, it is possible to obtain a much sharper error upper bound exploiting the results obtained by Brignone and Junike (2026). Suppose to have another approximation for $F_{X_T|V_T}$, denoted by $F^{\text{COS}}_{X_T|V_T}$, which is obtained following the same procedure as in (Brignone and Junike, 2026, Section 3.1). From (Brignone and Junike, 2026, Theorem 2), we have that $\left|F_{X_T|V_T}(x|v) - F^{\text{COS}}_{X_T|V_T}(x|v)\right| < \varepsilon$, where $\varepsilon > 0$ can be chosen arbitrarily small. Exploiting this result, and applying triangle inequality

we have from (22)

$$
\begin{aligned}
|P-\tilde{P}| &\leq S_0 e^{-rT}\int_0^{\infty} f_{V_T}(v)dv \int_{-\infty}^{\log\frac{K}{S_0}} e^x \Big| F_{X_T|V_T}(x|v) - F^{\text{COS}}_{X_T|V_T}(x|v)\Big| dx + \\
&+ S_0 e^{-rT}\int_0^{\infty} f_{V_T}(v)dv \int_{-\infty}^{\log\frac{K}{S_0}} e^x \Big| F^{\text{COS}}_{X_T|V_T}(x|v) - \tilde{F}_{X_T|V_T}(x|v)\Big| dx \\
&\leq K e^{-rT}\varepsilon + S_0 e^{-rT}\int_0^{\infty} f_{V_T}(v)dv \int_{-\infty}^{\log\frac{K}{S_0}} e^x \Big| F^{\text{COS}}_{X_T|V_T}(x|v) - \tilde{F}_{X_T|V_T}(x|v)\Big| dx.
\end{aligned}
$$

Note that the double integral in the above equation can be easily solved numerically since $F^{\text{COS}}_{X_T|V_T}$ is known full explicitly (Brignone and Junike, 2026, Formula 7). Therefore, the error upper bound for our proposed methodology is computable at least for the Heston model. Additionally, it is reasonably sharp. Indeed, setting $\varepsilon = 10^{-4}$, we obtained that $|P-\tilde{P}| < 0.0410$ for the parameter set H1. From a numerical point of view, to increase efficiency when solving the double integral it is convenient to transform the problem from the variance domain to the log-variance domain (see (Fang and Oosterlee, 2011, Section 2.3) for more details). Finally, let us stress that the true error is expected to be even smaller since in reality we are simulating $\left(X_T, \int_0^T V_s ds, V_T\right)$ and not $(X_T|V_T)$, as assumed when computing the error bound. Note that when simulating $\left(X_T, \int_0^T V_s ds, V_T\right)$ we expect higher accuracy (see Section EC.4). Despite the same approach can not be used to estimate error upper bounds for other polynomial models considered in this paper, the numerical results give confidence that such an error will be small also for such models. This is indeed what we found by estimating it numerically (see Table 4).

EC.9. Comparison with alternative simulation schemes

In Section 4 we have compared the performances of Algorithms 2–7 against the method of Kyriakou *et al.* (2024) (in the case of the Heston model) and against the Euler scheme. It is natural to ask how the proposed algorithms perform against alternative sampling schemes. In this section, we compare the performances of our Algorithm 4 against the interpolated Kahl and Jäckel (2006) method, that is the methodology used in Ackerer and Filipović (2020). Regarding the Heston model, the literature concerned with its simulation is massive, see, among many others, Andersen (2008), van Haastrecht and Pelsser (2010), Zhu (2011), Glasserman and Kim (2011), Tse and Wan (2013), Bégin *et al.* (2015), Okhrin *et al.* (2022), Choi and Kwok (2024). Among the various possible choices, we compare the performances of Algorithm 2 against the one proposed by Abi Jaber (2025).

EC.9.1. Comparison with the Interpolated Kahl and Jäckel method: the case of the Hull-White model

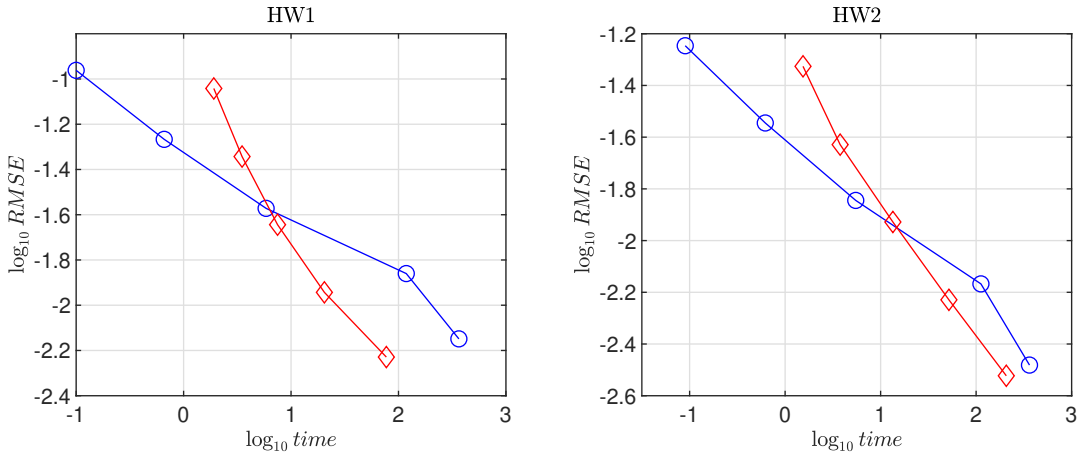
We compare our proposed Algorithm 4 against the Interpolated Kahl and Jäckel scheme. The choice of such benchmark scheme (instead of, for example, the Milstein scheme) is motivated with the fact that Ackerer and Filipović (2020) use an interpolated Kahl and Jäckel scheme to simulate polynomial stochastic volatility model. We start by running 10^{10} simulations of the HW model using the interpolated Kahl and Jäckel scheme (see (Ackerer and Filipović, 2020, p. 67)) for various number

Table 8: Speed-accuracy profiles of Algorithm 4 and the Interpolated Kahl and Jäckel scheme for parameter sets in Table 1.

	HW1						HW2					
	Interpolated Kahl-Jackel				Algorithm 4		Interpolated Kahl-Jackel				Algorithm 4	
$\mathcal{M}$	n	bias	RMSE	time	RMSE	time	n	bias	RMSE	time	RMSE	time
4×10^4	100	0.0598	0.1092	0.10	0.0908	1.91	100	0.0326	0.0567	0.09	0.0472	1.54
16×10^4	200	0.0299	0.0541	0.66	0.0454	3.5	200	0.0163	0.0285	0.62	0.0235	3.77
64×10^4	400	0.0144	0.0268	5.86	0.0227	7.49	400	0.0082	0.0143	5.49	0.0118	13.44
256×10^4	800	0.0079	0.0138	118.16	0.0114	20.43	800	0.0036	0.0068	112.30	0.0059	51.9
1024×10^4	1600	0.0044	0.0071	364.80	0.0059	77.09	1600	0.0016	0.0033	362.21	0.003	207.01

Notes. All computing times are expressed in seconds.

Figure 5: Speed-accuracy profiles of Algorithms 4 and the interpolated Kahl-Jackel scheme for parameter sets in Table 1 under the Advanced Hull-White stochastic volatility model.



Notes. Algorithm 4: plots with red diamond markers; Interpolated Kahl-Jackel scheme: plots with blue circle markers. All computing times are expressed in seconds.

of time steps $n = \{100, 200, 400, 800, 1600\}$. After running 10^{10} simulations, we compute European call option prices and evaluate the bias. Regressing $\log_{10} n$ vs $\log_{10} |\text{bias}|$ we obtain slopes very close to -1 for both HW1 and HW2. This means that the bias is approximately proportional to $1/(n^2)$. Duffie and Glynn (1995) show that if the bias decays at the order of $1/(n^p)$, then one should increase n proportionally to $\mathcal{M}^{1/(2p)}$ to achieve asymptotic optimality. Hence, having estimated $p \approx 1$ we can select n increasing at the order $\mathcal{N}^{1/2}$ (see also Cai *et al.* (2017)). RMSE and computing times are reported in Table 8 and (in log-log scale) in Figure 5. Results clearly indicate that Algorithm 4 performs better than the interpolated Kahl and Jäckel scheme, being faster for the same level of accuracy.

EC.9.2. Comparison with Abi Jaber (2025) under the Heston model

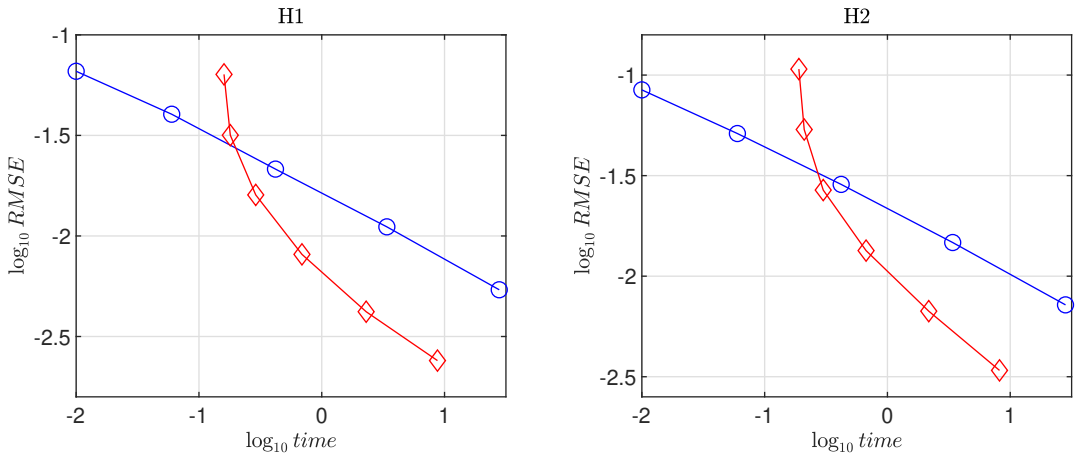
Abi Jaber (2025) provides a simulation scheme for the Heston model that is very fast, accurate, and simple to implement. The methodology is based on approximating the integrated variance using a moment-matched inverse Gaussian distribution and then on a straightforward right-endpoint Euler–type discretization rule. Such an approach has several merits. Mainly, it avoids that the variance becomes negative, which is a classical issue with the simulation of the Heston model. We run the methodology of Abi Jaber (2025) for 10^{10} simulations for various number of time steps

Table 9: Speed-accuracy profiles of Algorithm 2 and the method of Abi Jaber (2025) for parameter sets in Table 1

	H1						H2					
	Abi Jaber (2025)				Algorithm 2		Abi Jaber (2025)				Algorithm 2	
$\mathcal{M}$	n	bias	RMSE	time	RMSE	time	n	bias	RMSE	time	RMSE	time
1×10^4	1	0.0658	0.0969	0.00	0.0745	0.16	1	0.0716	0.1312	0.00	0.1136	0.19
4×10^4	2	0.0553	0.0659	0.01	0.0372	0.18	2	0.0635	0.0844	0.01	0.0565	0.21
16×10^4	4	0.0360	0.0403	0.06	0.0186	0.29	4	0.0429	0.0512	0.06	0.0283	0.3
64×10^4	8	0.0195	0.0215	0.42	0.0093	0.69	8	0.0249	0.0286	0.42	0.0142	0.67
256×10^4	16	0.0101	0.0111	3.39	0.0047	2.3	16	0.0129	0.0147	3.40	0.0071	2.17
1024×10^4	64	0.0049	0.0054	27.90	0.0025	8.76	64	0.0063	0.0072	28.20	0.0036	8.18

Notes. All computing times are expressed in seconds.

Figure 6: Speed-accuracy profiles of Algorithms 2 and the method of Abi Jaber (2025) for parameter sets in Table 1 under the Heston model.



Notes. Algorithm 2: plots with red diamond markers; method of Abi Jaber (2025): plots with blue circle markers. All computing times are expressed in seconds.

$n = \{1, 2, 4, 8, 16, 32\}$. For each level of n, we compute the bias. Then, we study the bias decay rate following the same approach described in Section EC.9.1 and select the number of time steps using the optimal allocation rule of Duffie and Glynn (1995). RMSE and computing times are reported in Table 9 and (in log-log scale) in Figure 6. Results clearly indicate that Algorithm 2 performs better than the benchmark method of Abi Jaber (2025), presenting an improved convergence rate of the RMSE. However, it should be noticed that our approach is designed for a single time step, while the method of Abi Jaber (2025) can efficiently handle multiple time steps and is thus expected to be more efficient when pricing frequently monitored path-dependent options.

EC.10. Speeding up Algorithm 3

In Section 4 we have presented the numerical performances of Algorithms 2–7 when pricing European options. We have noticed that in the case of the Jacobi model the computing times are higher than for the other models. This is because the built-in Matlab function `pearsrnd` that we used simulates from Pearson Type IV using an acceptance-rejection method based on a non optimal exponential envelope. However, it is possible to use more efficient envelopes. We recall the definition

of the Pearson's density

$$\frac{\partial f(Z)}{\partial Z} = -\frac{a_0 + Z}{b_0 + b_1 Z + b_{11} Z^2},$$

where the coefficients a_0, b_0, b_1, b_{11} are determined based on the mean μ, standard deviation σ, squared skewness β_1, and kurtosis β_2 of Z. In what follows, let us suppose, without loss of generality, that $\mu = 0$ and $\sigma = 1$. In the case of Type IV, we have that the probability density function of Z is given as follows

$$f(z) = C \left(1 + \left(\frac{z-\lambda}{a}\right)^2\right)^{-m} \exp\left(-\nu \arctan\left(\frac{z-\lambda}{a}\right)\right),$$

where C is a normalizing constant defined below and where

$$m = \frac{1}{2b_{11}},\ \nu = \frac{2b_1(1-m)}{\sqrt{4b_0 b_{11} - b_1^2}},\ b = 2(m-1),\ a = \sqrt{\frac{b^2(b-1)}{b^2+\nu^2}},\ \lambda = \frac{a\nu}{b}.$$

Setting $a_\star = m$ and $s_\star = -\nu$, the probability density function of $Y = \frac{Z-\lambda}{a}$ is characterized by two shape parameters, $a_\star > 1/2$ and $s_\star \in \mathbb{R}$, has density on the real line given by

$$f(y) = \frac{C e^{s_\star \arctan(y)}}{(1+y^2)^{a_\star}},\ \text{where } C = \frac{4^{a_\star - 1} |\Gamma(a_\star - i s_\star/2)|^2}{\pi \Gamma(2a_\star - 1)}.$$

For some values of $a_\star$ and $s_\star$ it is possible to improve the random sampling scheme for the Pearson Type IV. In particular, Devroye and Hill (2025, Section 9) suggest the following improvement for simulating Y:

i) For $a_\star \geq 1$ and $\forall s_\star$, one can use the universal log-concave generator by Heinrich (2004). This is what is used in Section 4.

ii) For $s_\star \leq 5$ and $\forall a_\star$, one can use rejection from the Student-t density.

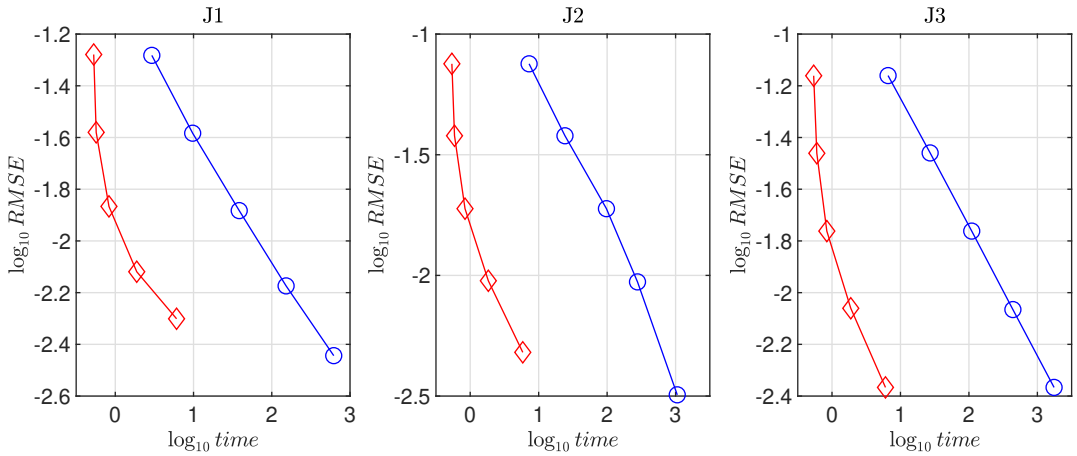
iii) For $a_\star \in (1/2, 1]$ and $s_\star \geq 1$, one can use rejection from the gamma density after symmetrizing the Pearson distribution.

iv) For $a_\star = 1$, it is possible to sample from a skewed Cauchy density (that is straightforward, see Devroye and Hill, 2025)

v) For $s_\star = 0$, it is possible to sample from a Student-t law, as explained below.

The relevant case in our context is the fifth. Indeed, under the Jacobi model we have that the skewness of $\left(X_T | V_T, \int_0^T V_s ds\right)$ is very close to zero, rendering the number of rejected samples huge, and the algorithm slow, when applying the universal log-concave generator of Heinrich (2004). In such cases, $s_\star$ is very close to 0. Therefore, it is possible to avoid the universal log-concave generator by Heinrich (2004) and simulate directly from a Student-t distribution, accelerating computations. More

Figure 7: Speed-accuracy profiles of Algorithms 3 and its modification based on the results in Devroye and Hill (2025) for parameter sets in Table 1 under the Jacobi stochastic volatility model.



Notes. Modified Algorithm 3: plots with red diamond markers; Algorithm 3: plots with blue circle markers. All computing times are expressed in seconds.

specifically, when $s_\star = 0$, following Devroye and Hill (2025), a random sample from the Pearson Type IV distribution can be generated by sampling two independent uniforms U' and U, and by computing

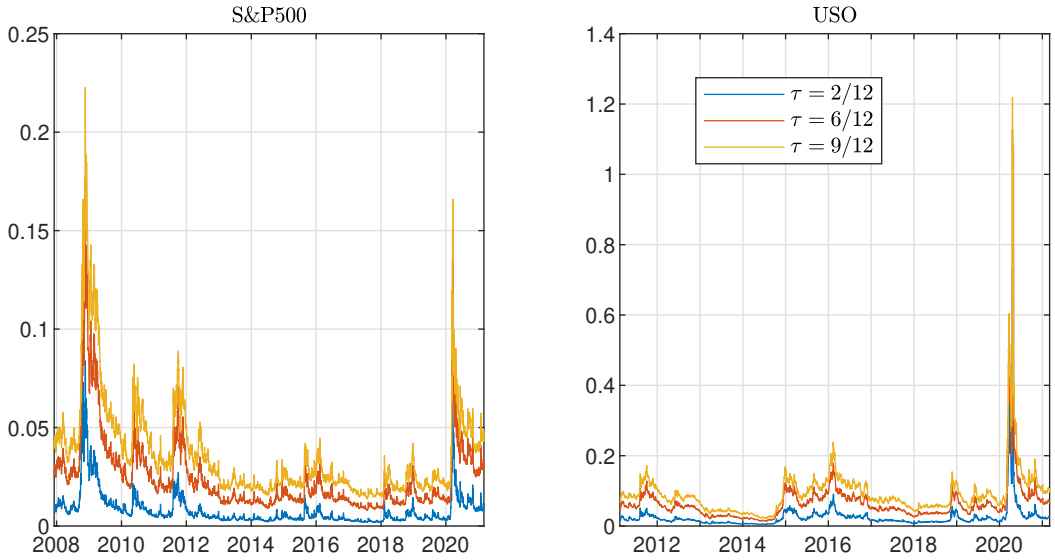
$$\sin\left(2\pi U'\right)\sqrt{\frac{1}{U^{2/a_\star}}-1}. \tag{15}$$

The problem of simulating using the method of Heinrich (2004) from the Pearson Type IV with $s_\star \approx 0$ arises only under the Jacobi model, where, indeed, numerical performances were worse than for other models. However, in this case it is possible to improve the performances. In particular, we modify Algorithm 3 as follows. When Pearson Type IV is selected and when $s_\star \approx 0$ (in practice, when $|s_\star| < 10^{-5}$), we sample from (15). Of course, the resulting algorithm will be much faster. What about the accuracy? We run 10^{10} simulations of Algorithm 3 with the aforementioned modification under the same settings specified in Section 4. We obtain the following prices for the European call options: 7.6078 for J1, for 10.2255 for J2, and 8.4364 for J3. Therefore, as expected, the bias is a bit larger than that obtained with the universal log-concave generator (compare with Table 4). However, the algorithm is much faster. Indeed, for 256×10^4 simulations, the computing time in seconds becomes 1.88 seconds for J1, 0.84 seconds for J2, and 0.84 seconds for J3. Comparing with the results reported in Table 6, we observe that the running time is reduced by more than 97% in all the parameter sets considered. Speed-accuracy profiles for Algorithm 3 against its modification proposed in this section are reported in Figure 7.

EC.11. Estimation of stochastic volatility models under both historical and risk-neutral measures

In this section, we estimate the Jacobi and advanced Hull-White models on time series of derivatives prices by exploiting statistical filtering methods. The methodology we are going to use is subject to many limitations and we point out that our scope is just to obtain realistic parameter

Figure 8: Time series of expected realized variances computed as in (16) for three different maturities.



values which are used to test the performances of our simulation method for the various models. The computational econometric literature is indeed more focused on affine option pricing models since efficient pricing formulas for European options are available. To the best of our knowledge, no general econometric methodology is available for option pricing models based on polynomial processes which uses derivatives data. Developing such procedure is far beyond the scope of this work and is currently under consideration for future research.

Under mild assumptions, the expected value of the future realized variance can be computed, following Jiang and Tian (2005, 2007), as

$$\mathrm{Var}_t^{\mathrm{mkt}}(T) = \mathrm{E}\left[\int_t^T V_s ds \Big| \mathcal{F}_t\right] = 2\int_0^\infty \frac{C_t(T-t, e^{-r(T-t)}K) - \max(0, S_t - K)}{K^2} dK \qquad (16)$$

where $C_t(T,K)$ is the price of an option at time t with maturity $\tau = T - t$ and strike price K. Therefore, given a time series of quoted option prices, we can build the time series of market implied expected realized variance, $\mathrm{Var}_t(T)$, for various maturities. We consider time series of option prices for S&P500 and USO available from Refinitiv Eikon and build the daily time series of expected realized variances at various maturities. For the S&P our dataset ranges from December 5, 2007 to March 4, 2021, for the USO it ranges from February 16, 2011 to March 4, 2021. Implementing (16) is not straightforward due to the limited availability of the strikes in the market. We implement the fast and stable methodology described in Jackwerth (2004) to smooth the implied volatility surface and obtain accurate results for $\mathrm{Var}_t(T)$. The resulting time series is reported in Figure 8 for various maturities $\tau = T - t = \{2, 6, 9\}$ months.

To estimate the models on derivatives data we have to compute model implied expected realized variance. This is straightforward under polynomial models. Indeed, from the polynomial property we can always compute the moments of $\int_0^T V_s ds$. Before doing this, we have to define the risk-neutral

parameters. We follow the standard approach (see Fulop and Li (2019)) and obtain the following relationship between parameters under the historical and risk-neutral measures:

$$\begin{cases} k_{\mathbb{Q}} = k + \sigma\phi, \quad \theta_{\mathbb{Q}} = k\theta/k_{\mathbb{Q}}, & \text{Jacobi} \\ k_{\mathbb{Q}} = k + \phi, \quad \theta_{\mathbb{Q}} = k\theta/k_{\mathbb{Q}}, & \text{Advanced Hull-White} \end{cases}$$

where ϕ denotes the risk-premium. Applying the moment formula we obtain the model implied expected risk-neutral realized variance:

$$\text{Var}_t(\tau, \Theta) = \text{E}\left[\int_t^T V_s ds \Big| \mathcal{F}_t\right] = A(\tau) + B(\tau)V_t \tag{17}$$

where Θ denote the vector of the model parameters, $A(\tau) = (\theta_{\mathbb{Q}} \exp(-k_{\mathbb{Q}}) - \theta_{\mathbb{Q}} + k_{\mathbb{Q}}\theta_{\mathbb{Q}})/k_{\mathbb{Q}}$ and $B(\tau) = -(\exp(-\tau k_{\mathbb{Q}}) - 1)/k_{\mathbb{Q}}$ for the Jacobi model and

$$\text{Var}_t(\tau, \Theta) = \text{E}\left[\int_t^T V_s ds \Big| \mathcal{F}_t\right] = A(\tau) + B(\tau)Y_t + C(\tau)V_t \tag{18}$$

where

$$\begin{aligned} A(\tau) &= \frac{2}{(2\,k_{\mathbb{Q}} - \sigma^2)^2\ (k_{\mathbb{Q}} - \sigma^2)}\,{\theta_{\mathbb{Q}}}^2 \Big(2\,\tau\,k_{\mathbb{Q}}^3 - k_{\mathbb{Q}}^2\,\mathrm{e}^{\tau\,\sigma^2 - 2\,\tau\,k_{\mathbb{Q}}} + 4\,\sigma^2\,k_{\mathbb{Q}} + \sigma^4\,\mathrm{e}^{-\tau\,k_{\mathbb{Q}}} + 4\,k_{\mathbb{Q}}^2\,\mathrm{e}^{-\tau\,k_{\mathbb{Q}}} - \sigma^4 + \\ &\quad - 3\,k_{\mathbb{Q}}^2 - 4\,\sigma^2\,k_{\mathbb{Q}}\,\mathrm{e}^{-\tau\,k_{\mathbb{Q}}} - 3\,\tau\,\sigma^2\,k_{\mathbb{Q}}^2 + \tau\,\sigma^4\,k_{\mathbb{Q}}\Big), \\ B(\tau) &= \frac{2\,\theta_{\mathbb{Q}}\,\left(k_{\mathbb{Q}} + k_{\mathbb{Q}}\,\mathrm{e}^{\tau\,\sigma^2 - 2\,\tau\,k_{\mathbb{Q}}} - 2\,k_{\mathbb{Q}}\,\mathrm{e}^{-\tau\,k_{\mathbb{Q}}} + \sigma^2\,\mathrm{e}^{-\tau\,k_{\mathbb{Q}}} - \sigma^2\right)}{(2\,k_{\mathbb{Q}} - \sigma^2)\ (k_{\mathbb{Q}} - \sigma^2)}, \\ C(\tau) &= -\frac{\mathrm{e}^{\tau\,\sigma^2 - 2\,\tau\,k_{\mathbb{Q}}} - 1}{2\,k_{\mathbb{Q}} - \sigma^2}, \end{aligned}$$

for the Advanced Hull-White model. Therefore, we estimate the models using a two steps optimization. First, we run the CSIR algorithm outlined in Section EC.1. Given model parameters under the historical measure and the filtered variance, we compute $\text{E}\left[\int_t^T V_s ds|\mathcal{F}_t\right]$ using formulas (17)-(18) and compute the mean squared errors with respect to market values:

$$\text{MSE}(\Theta) = \frac{1}{n}\sum_{t=1}^{n_T}\sum_{j=1}^{3}(\text{Var}_t(\tau_j)^{\text{mkt}} - \text{Var}_t(\tau_j, \Theta))^2, \tag{19}$$

for $\tau_1 = 2/12$, $\tau_2 = 6/12$, $\tau_3 = 9/12$. The estimated parameters are those minimizing the MSE computed in this way. The numerical minimization is performed using the built-in Matlab function `fmincon`. Results are reported in Table 10.

EC.12. Estimation of the Pilipović model

Next, we estimate the Pilipović model exploiting time series of futures prices. The scope is obtaining realistic model parameters to test the simulation methodology proposed in this paper. For

Table 10: Joint estimation under historical and risk-neutral measures.

	Jacobi			Advanced Hull-White	
Θ	S&P 500	USO	Θ	S&P 500	USO
μ	0.0869	-0.1726	μ	0.0676	-0.1675
k	1.1674	1.6389	k	1.3577	2.3775
θ	0.0719	0.1521	θ	0.2397	0.3540
σ	0.2898	0.6590	σ	0.4578	0.8861
ρ	-0.2324	-0.6409	ρ	-0.4635	-0.6434
v_{min}	0.0017	0.0029	$k_{\mathbb{Q}}$	1.1725	2.3775
v_{max}	0.7723	1.2743	$\theta_{\mathbb{Q}}$	0.2775	0.3540
$k_{\mathbb{Q}}$	0.9329	1.5680			
$\theta_{\mathbb{Q}}$	0.0899	0.1590			

simplicity, let us assume that there are no risk premia, i.e. the parameters can be estimated directly from time series of futures prices without including observations under the historical measure. For model estimation, we consider the time series of Crude Oil WTI futures quotes from December 19, 2003 to August 5, 2022 and three different maturities $\tau = \{1, 3, 6\}/12$ years. Thus, for a given day t we observe the price of future contracts written on the West Texas Intermediate (WTI) at $J = 3$ different maturities, denoted by $\hat{F}_t(\tau)$.

Under the Pilipović model the price of a future at time t with maturity τ can be computed as follows

$$F_t(\tau) = \mathrm{E}[S_{t+\tau}|\mathcal{F}_t] = \exp(-k\tau)S_t + \frac{k\,\mathrm{e}^{\tau\,\eta} - k\,\mathrm{e}^{-\tau\,k}}{\eta + k}L_t. \tag{20}$$

We cast the Pilipović model (20)–(19) into a discrete time state-space model. Let us define $c_t = F_t(\tau)^\top$ and $c_t^{\mathrm{mkt}} = \hat{F}_t(\tau)^\top$. Equation (20) implies that c_t is linearly related to the 2×1 vector of latent factors $H_t = (S_t, L_t)^\top$. Thus, the measurement equation is

$$c_t = \Gamma_1 F_t + \eta_t, \tag{21}$$

where $\Gamma_1 = \begin{bmatrix}\exp(-k\tau) & \frac{k\,\mathrm{e}^{\tau\,\eta} - k\,\mathrm{e}^{-\tau\,k}}{\eta+k}\end{bmatrix}$ is a $J \times 2$ matrix of coefficients . Further, $\eta_t \sim \mathcal{N}(0, \Omega)$ where Ω is a $J \times J$ diagonal covariance matrix. The transition equation is obtained from an Euler approximation with step size Δt of the continuous-time model (20)–(19) and it is given as

$$H_t = \Phi_1 H_{t-1} + \varepsilon_t, \tag{22}$$

where Φ_1 is a 2×2 matrix of coefficients. The transition noise is $\varepsilon_t \sim \mathcal{N}(0, \Sigma(F_{t-1}))$, where ε_t is independent of ζ_t and $\Sigma(F_{t-1}) = \mathrm{Cov}(F_{t-1})$ is the conditional covariance of F_{t-1}. The expression for Φ_1 is

$$\Phi_1 = I_2 + K_1, \quad K_1 = \begin{bmatrix} -k\Delta t + 1 & k\Delta t \\ 0 & 1 + \eta\Delta t \end{bmatrix}, \tag{23}$$

where I_2 is a 2×2 identity matrix. The transition noise has the following conditional covariance

Table 11: Parameter estimates for the Pilipović model.

Θ	k	σ	η	γ	ρ
Estimate	1.2596	0.3632	-0.1077	0.2532	0.7257
s.e.	0.0755	0.0637	0.1166	0.0047	0.1248

matrix

$$\Sigma(F_{t-1}) = \mathrm{Cov}(F_{t-1}) = \Delta t \begin{bmatrix} \sigma^2 S_{t-1}^2 & \gamma\rho\sigma L_{t-1} S_{t-1} \\ \gamma\rho\sigma L_{t-1} S_{t-1} & \gamma^2 L_{t-1}^2 \end{bmatrix}. \tag{24}$$

The standard Kalman filter is not optimal in this setting because the conditional covariance $\mathrm{Cov}(F_{t-1})$ depends on the state itself. Hence, we are implementing a modified Kalman filter, where the transition noise at time $t-1$ is used as an estimate for the time t iteration. If we define $H_{t|t} = \mathrm{E}[H_t]$ and $P_{t|t} = \mathrm{Cov}[H_t]$, the recursions of the filter are

$$\begin{cases} H_{t+1|t} = \Phi_1 H_{t|t} \\ P_{t+1|t} = \Phi_1 P_{t|t} \Phi_1^\top + \Sigma(H_{t|t}) \\ c_{t+1|t} = \Gamma_0 + \Gamma_1 H_{t+1|t} \\ M_{t+1|t} = \Gamma_1 P_{t+1|t} \Gamma_1^\top + \Omega \\ H_{t+1|t+1} = \max\left(H_{t+1|t} + P_{t+1|t}\Gamma_1^\top M_{t+1|t}^{-1}\left(c_{t+1}^{\mathrm{mkt}} - c_{t+1|t}\right), 0\right) \\ P_{t+1|t+1} = P_{t+1|t} - P_{t+1|t}\Gamma_1^\top M_{t+1|t}^{-1}\Gamma_1 P_{t+1|t} \end{cases} \tag{25}$$

and the Gaussian quasi log-likelihood is given by

$$-\frac{1}{2}\sum_{t=1}^{T} \ln\left((2\pi)^{3J}\det(M_{t|t-1})\right) + \left(c_t^{\mathrm{mkt}} - c_{t|t-1}\right)^\top M_{t|t-1}^{-1}\left(c_t^{\mathrm{mkt}} - c_{t|t-1}\right). \tag{26}$$

The properties of the modified Kalman filter are discussed in Monfort *et al.* (2017) (see also Brignone *et al.* (2023)). We can now estimate the model by numerically maximizing the quasi log-likelihood given in (26). The estimated parameters and related standard errors are reported in Table 11. Additionally, we report in Figure 9 the filtered price and level. They both show realistic patterns. Finally, we show in Figure 10 the time series of market (red line) and model (blue line) implied futures prices. Results show that the calibrated models realistically reproduce the time series of futures prices.

Figure 9: Filtered latent states.

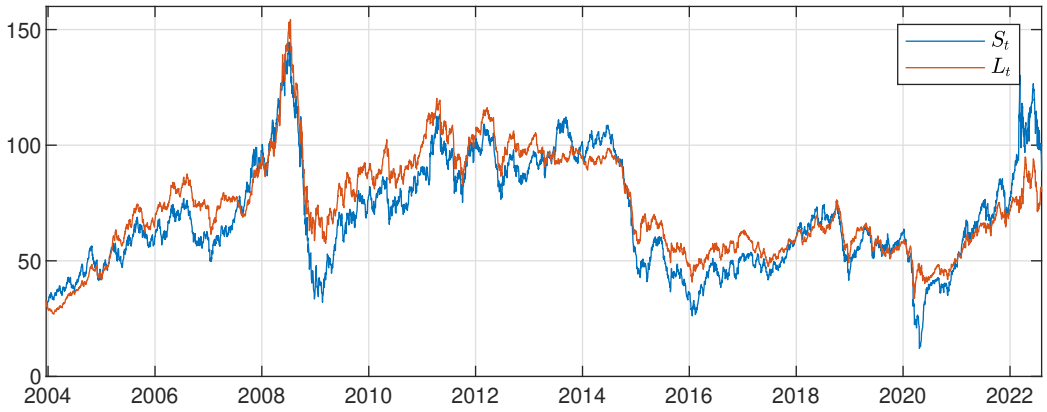
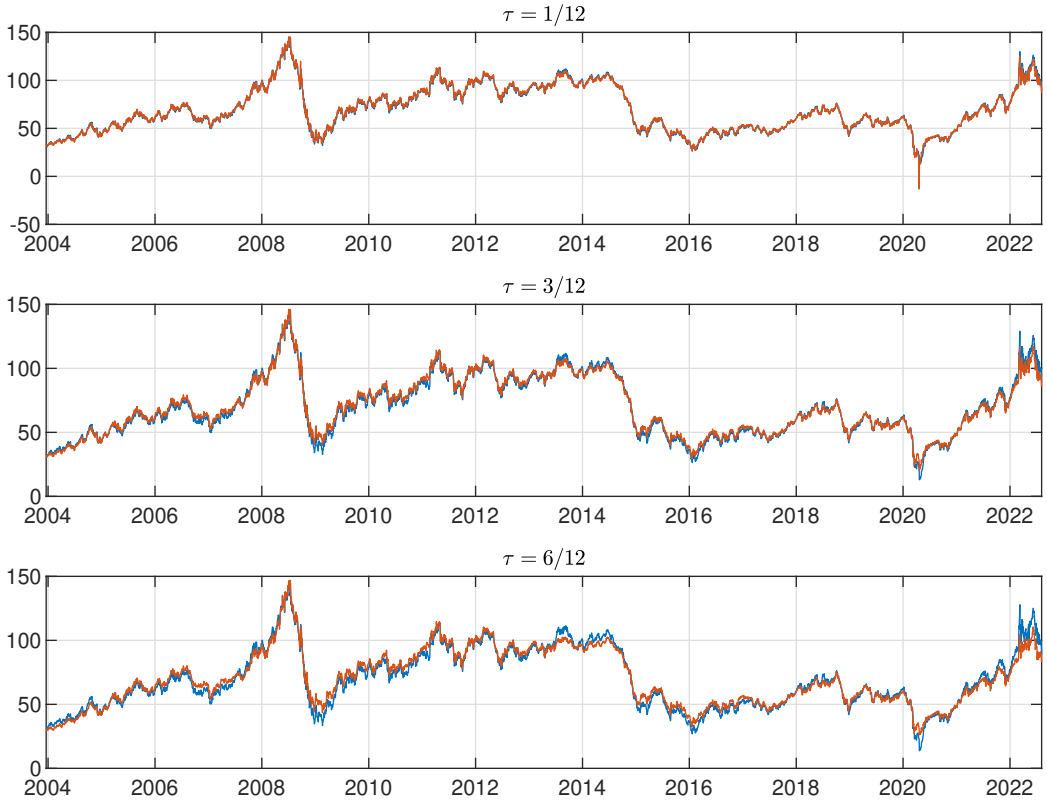


Figure 10: Time series of market vs model futures prices in correspondence of the estimated parameters for various maturities.



Notes. Blue line denote the model implied futures prices, red lines denote the market implied futures prices.

E-companion References

Abi Jaber, E. (2025) Simulation of square-root processes made simple: applications to the Heston model. *Risk Magazine*.

Ackerer, D. and Filipović, D. (2020) Option pricing with orthogonal polynomial expansions. *Mathematical Finance*, **30**, 47–84.

Andersen, L. (2008) Simple and efficient simulation of the Heston stochastic volatility model. *Journal of Computational Finance*, **11**, 1–42.

Arbel, J., Lijoi, A. and Nipoti, B. (2016) Full Bayesian inference with hazard mixture models. *Computational Statistics & Data Analysis*, **93**, 359–372.

Bégin, J., Bédard, M. and Gaillardetz, P. (2015) Simulating from the Heston model: A gamma approximation scheme. *Monte Carlo Methods and Applications*, **21**, 205–231.

Biagini, F. and Zhang, Y. (2016) Polynomial diffusion models for life insurance liabilities. *Insurance: Mathematics and Economics*, **71**, 114–129.

Brennan, M. and Schwartz, E. S. (1980) Analyzing convertible bonds. *Journal of Financial and Quantitative Analysis*, **15**, 907–929.

Brignone, R. (2024) Exact simulation of the multifactor Ornstein–Uhlenbeck driven stochastic volatility model. *SIAM Journal on Scientific Computing*, **46**, A1441–A1460.

Brignone, R. and Gonzato, L. (2024) Exact simulation of the Hull and White stochastic volatility model. *Journal of Economic Dynamics and Control*, **163**, 104861.

Brignone, R., Gonzato, L. and Lütkebohmert, E. (2023) Efficient quasi-Bayesian estimation of affine option pricing models using risk-neutral cumulants. *Journal of Banking and Finance*, **148**, 106745.

Brignone, R. and Junike, G. (2026) Exact simulation of stochastic volatility models based on conditional Fourier-cosine method. *European Journal of Operational Research*, **328**, 1036–1053.

Cai, N., Li, C. and Shi, C. (2014) Closed-form expansions of discretely monitored Asian options in diffusion models. *Mathematics of Operations Research*, **39**, 789–822.

Cai, N., Song, Y. and Chen, N. (2017) Exact simulation of the SABR model. *Operations Research*, **65**, 931–951.

Choi, J. and Kwok, Y. (2024) Simulation schemes for the Heston model with Poisson conditioning. *European Journal of Operational Research*, **314**, 363–376.

Christoffersen, P., Heston, S. and Jacobs, K. (2009) The shape and term structure of the index option smirk: Why multifactor stochastic volatility models work so well. *Management Science*, **55**, 1914–1932.

Devroye, L. and Hill, J. (2025) Simulating random variates from the Pearson IV and betaized Meixner–Morris distributions. *Available at* ***https://arxiv.org/abs/2505.15930***.

Doucet, A. and Johansen, A. M. (2008) A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, **12**, 656–704.

Duffie, D. and Glynn, P. (1995) Efficient Monte Carlo estimation of security prices. *Annals of Applied Probability*, **4**, 897–9058.

Fang, F. and Oosterlee, C. (2011) A Fourier-based valuation method for Bermudan and barrier options under Heston's model. *SIAM J. Financial Math.*, **2**, 439–463.

Filipović, D., Mayerhofer, E. and Schneider, P. (2013) Density approximations for multivariate affine jump-diffusion processes. *Journal of Econometrics*, **176**, 93–111.

Fulop, A. and Li, J. (2019) Bayesian estimation of dynamic asset pricing models with informative observations. *Journal of Econometrics*, **209**, 114–138.

Fusai, G. and Tagliani, A. (2002) An accurate valuation of Asian options using moments. *International Journal of Theoretical and Applied Finance*, **5**, 147–169.

Glasserman, P. and Kim, K. K. (2011) Gamma expansion of the Heston stochastic volatility model. *Finance and Stochastics*, **15**, 267–296.

Heinrich, J. (2004) A guide to the Pearson Type IV distribution. Technical report.

Hubalek, F., Keller-Ressel, M. and Sgarra, C. (2017) Geometric Asian option pricing in general affine stochastic volatility models with jumps. *Quantitative Finance*, **17**, 873–888.

Jackwerth, J. C. (2004) Option-implied risk-neutral distributions and risk aversion. *Research Foundation of AIMR, CFA Institute*.

Jiang, G. J. and Tian, Y. S. (2005) The model-free implied volatility and its information content. *The Review of Financial Studies*, **18**, 1305–1342.

Jiang, G. J. and Tian, Y. S. (2007) Extracting model-free volatility from option prices: An examination of the VIX index. *Journal of Derivatives*, **14**, 35–60.

Kahl, C. and Jäckel, P. (2006) Fast strong approximation Monte Carlo schemes for stochastic volatility models. *Quantitative Finance*, **6**, 513–536.

Kloeden, P. E. and Platen, E. (1992) *Numerical Solution of Stochastic Differential Equations*. Springer-Verlag Berlin Heidelberg.

Kyriakou, I., Brignone, R. and Fusai, G. (2024) Unified moment–based modelling of integrated stochastic processes. *Operations Research*, **72**, 1630–1653.

Larsson, M. and Pulido, S. (2017) Polynomial preserving diffusions on compact quadric sets. *Stochastic Processes and their Applications*, **127**, 901–926.

Li, M., Mercurio, F. and Resnick, S. (2018) The Garch linear SDE: Explicit formulas and the pricing of a quanto CDS. *Risk*.

Lindsay, B. G. and Basak, P. (2000) Moments determine the tail of a distribution (but not much else). *The American Statistician*, **54**, 248–251.

Malik, S. and Pitt, M. K. (2011) Particle filters for continuous likelihood evaluation and maximisation. *Journal of Econometrics*, **165**, 190–209.

Matsumoto, H. and Yor, M. (2005) Exponential functionals of Brownian motion 1: Probability laws at fixed time. *Probability Surveys*, **2**, 312–347.

Monfort, A. Pegoraro, F., Renne, J. P. and Rousselet, G. (2017) Staying at zero with affine processes: An application to term structure modelling. *Journal of Econometrics*, **201**, 348–366.

Okhrin, O., Rockinger, M. and Schmid, M. (2022) Simulating the Cox-Ingersoll-Ross and Heston processes: matching the first four moments. *Journal of Computational Finance*, **26**, 1–52.

Provost, S. (2005) Moment-based density approximants. *The Mathematica Journal*, **9**, 727–756.

Recchioni, M., Iori, G., Tedeschi, G. and Ouellette, M. (2021) The complete Gaussian kernel in the multi-factor Heston model: Option pricing and implied volatility applications. *European Journal of Operational Research*, **293**, 336–360.

Tse, T. and Wan, J. (2013) Low-bias simulation scheme for the Heston model by Inverse Gaussian approximation. *Quantitative Finance*, **16**, 919–937.

van Haastrecht, A. and Pelsser, A. (2010) Efficient, almost exact simulation of the Heston stochastic volatility model. *International Journal of Theoretical and Applied Finance*, **13**, 1–43.

Zhao, C., van Beek, M., Spreij, P. and Ba, M. (2025) Polynomial approximation of discounted moments. *Finance and Stochastics*, **29**, 63–95.

Zhu, J. (2011) A simple and accurate simulation approach to the heston model. *Journal of Derivatives*, **18(4)**, 1–43.