Proceedings of the 53rd ISCIE International Symposium
on Stochastic Systems Theory and Its Applications
Kusatsu, Oct. 30-31, 2021

An Algorithm for the Computation of Joint Hawkes Moments with Exponential Kernel[*]

Nicolas Privault
School of Physical and Mathematical Sciences,
Nanyang Technological University
21 Nanyang Link, Singapore 637371
E-mail: `nprivault@ntu.edu.sg`

Abstract

The purpose of this paper is to present a recursive algorithm and its implementation in Maple and Mathematica for the computation of joint moments and cumulants of Hawkes processes with exponential kernels. Numerical results and computation times are also discussed. Obtaining closed form expressions can be computationally intensive, as joint fifth cumulant and moment formulas can be respectively expanded into up to 3,288 and 27,116 summands.

1 Introduction

Hawkes processes [Haw71] are self-exciting point processes that have found applications in fields such as earth science with the study of earthquakes [Oga98], crime data [MSB+11], infectious diseases [SHH19], neuroscience [CR10], genomics analysis [RBS10], as well as finance [ELL11], or social media [RLMX18].

The analysis of statistical properties of Hawkes processes is made difficult by their recursive nature, making the computation of moments difficult. In [JHR15], a tree-based method for the computation of cumulants has been introduced, with an explicit computation of third order cumulants. Ordinary differential equation (ODE) methods have been applied in [DZ11] to the computation of the moment and probability generating functions of (generalized) Hawkes processes and their intensity, with the computation of first and second moments in the stationary case, see also [EGG10], [CHY20], and [DP20] for other ODE-based approaches.

In [BDM12], stochastic calculus and martingale arguments have been applied to the computation of first and second order moments, however those approaches seem difficult to generalize to higher-orders moments. In [LRV], cumulant recursion formulas have been obtained for general random variables using martingale brackets. Third-order cumulant expressions for Hawkes processes have been used in [ABMR18] for the analysis of order books in finance, and in [OJSBB17], [MMG20] for neuronal networks. Multivariate cumulants have been used for Hawkes kernel estimation in [ABG+17], [YSB21].

In [Pri21], the cumulants of Hawkes processes have been computed using using Bell polynomials, based on a recursive relation for the Probability Generating Functional (PGFl) of self-exciting point processes started from a single point. This provides a closed-form alternative to the tree-based approach of [JHR15].

In this note we apply the algorithm of [Pri21] to the recursive computation of joint moments of all orders of Hawkes processes, and present the corresponding codes written in Maple and Mathematica. The algorithm uses sums over partitions and Bell polynomials to compute joint cumulants in the case of an exponential branching intensity on $[0,\infty)$, and does not rely on differential equations.

We proceed as follows. After reviewing some combinatorial identities in Section 2, we will consider the computation of the joint cumulants of self-exciting Hawkes Poisson cluster processes in Section 3. Explicit computations for the time-dependent joint third and fourth cumulants of Hawkes processes with exponential kernels are presented in Section 4, and are confirmed by Monte Carlo estimates.

2 Joint moments and cumulants

In this section we present background combinatorial results that will be needed in the sequel. Given the Moment Generating Function (MGF)

$$\begin{aligned} M_X(t_1,\ldots,t_n) &:= \mathbb{E}\big[\mathrm{e}^{t_1X_1+\cdots+t_nX_n}\big] \\ &= 1+\sum_{k_1,\ldots,k_n\geq 1}\frac{t_1^{k_1}\cdots t_n^{k_n}}{n!}\mathbb{E}[X_1^{k_1}\cdots X_n^{k_n}], \end{aligned}$$

of a random vector $X=(X_1,\ldots,X_n)$, the joint cumulants of $(X_1,\ldots,X_n)$ of orders $(l_1,\ldots,l_n)$ are the coefficients $\kappa_{l_1,\ldots,l_n}(X_1,\ldots,X_n)$ appearing in the log-MGF

[*]This research is supported by the Ministry of Education, Singapore, under its Tier 2 Grant MOE-T2EP20120-0005.

expansion

$$\log M_X(t_1,\ldots,t_n) = \log\big(\mathbb{E}\big[\mathrm{e}^{t_1X_1+\cdots+t_nX_n}\big]\big) \qquad (1)$$
$$= \sum_{l_1,\ldots,l_n\geq 1} \frac{t_1^{l_1}\cdots t_n^{l_n}}{l_1!\cdots l_n!}\kappa_{l_1,\ldots,l_n}(X_1,\ldots,X_n),$$

for $(t_1,\ldots,t_n)$ in a neighborhood of zero in $\mathbb{R}^n$. In the sequel we let

$$\kappa(X_1,\ldots,X_n) := \kappa_{1,\ldots,1}(X_1,\ldots,X_n), \quad n\geq 1,$$

and

$$\kappa^{(n)}(X) := \kappa_{1,\ldots,1}(X,\ldots,X), \quad n\geq 1.$$

The joint moments of $(X_1,\ldots,X_n)$ are then given by the joint moment-cumulant relation

$$\mathbb{E}[X_1\cdots X_n] = \sum_{l=1}^n \sum_{\pi_1\cup\cdots\cup\pi_l=\{1,\ldots,n\}} \prod_{j=1}^l \kappa^{(|\pi_j|)}\big((X_i)_{i\in\pi_j}\big).$$

where the sum runs over the partitions $\pi_1,\ldots,\pi_k$ of the set $\{1,\ldots,n\}$. By the multivariate Faà di Bruno formula, (1) can be inverted as

$$\kappa(X_1,\ldots,X_n)$$
$$= \sum_{l=1}^n (l-1)!(-1)^{l-1} \sum_{\pi_1\cup\cdots\cup\pi_l=\{1,\ldots,n\}} \prod_{j=1}^l \mathbb{E}\left[\prod_{i\in\pi_j} X_i\right].$$

In the univariate case, the moments $\mathbb{E}[X^n]$ of a random variable X are linked to its cumulants $\big(\kappa^{(n)}(X)\big)_{n\geq 1}$ through the relation

$$\mathbb{E}[X^n] = B_n\big(\kappa^{(1)}(X),\ldots,\kappa^{(n)}(X)\big)$$
$$= \sum_{k=1}^n B_{n,k}(\kappa^{(1)}(X),\ldots,\kappa^{(n-k+1)}(X)),$$

where

$$B_{n,k}(a_1,\ldots,a_{n-k+1}) = \frac{n!}{k!} \sum_{\substack{l_1+\cdots+l_k=n\\ l_1\geq 1,\ldots,l_k\geq 1}} \frac{a_{l_1}}{l_1!}\cdots\frac{a_{l_k}}{l_k!}$$
$$= \sum_{\pi_1\cup\cdots\cup\pi_k=\{1,\ldots,n\}} a_{|\pi_1|}(X)\cdots a_{|\pi_k|}(X), \qquad (2)$$

$1\leq k\leq n$, is the partial Bell polynomial of order (n,k), where the sum (2) holds on the integer compositions $(l_1,\ldots,l_k)$ of n, see e.g. Relation (2.5) in [Luk55], and

$$B_n(a_1,\ldots,a_n) = \sum_{k=1}^n B_{n,k}(a_1,\ldots,a_{n-k+1})$$

is the complete Bell polynomial of degree $n\geq 1$. We also have the inversion relation

$$\kappa^{(n)}(X)$$
$$= \sum_{k=0}^{n-1} k!(-1)^k B_{n,k+1}\big(\mathbb{E}[X],\mathbb{E}[X^2],\ldots,\mathbb{E}[X^{n-k}]\big),$$

$n\geq 1$, see e.g. Theorem 1 of [Luk55], and also [LS59], Relations (2.8)-(2.9) in [McC87], or Corollary 5.1.6 in [Sta99].

As an example, we consider the recursive computation of Borel cumulants as an example. Let $(X_n)_{n\geq 0}$ be a branching process started at $X_0 = 1$ with Poisson distributed offspring count N of parameter $\mu\in(0,1)$, and let X denote the total count of offsprings generated by $(X_n)_{n\geq 0}$ It is known, see [PS98] and § 3.2 of [CF06] that X has the Borel distribution

$$\mathbb{P}(X=n) = \mathrm{e}^{-\mu n}\frac{(\mu n)^{n-1}}{n!}, \qquad n\geq 1.$$

We have $\kappa^{(1)}(X) = 1/(1-\mu)$ and the induction relation

$$\kappa^{(n)}(X) = \frac{\mu}{1-\mu}\big(B_n\big(\kappa^{(1)}(X),\ldots,\kappa^{(n)}(X)\big) - \kappa^{(n)}(X)\big)$$
$$= \frac{\mu}{1-\mu}\sum_{k=2}^n B_{n,k}\big(\kappa^{(1)}(X),\ldots,\kappa^{(n-k+1)}(X)\big), \qquad (3)$$

$n\geq 2$, see § 8.4.3 in [CF06] and Proposition 2.1 in [Pri21]. The recursion (3) is implemented in the following Maple code.

```
c := proc(n, mu) local tmp, k, z1; option remember; if n = 1 then
      return 1/(1 - mu); end if;
  tmp := 0; z1 := []; for k from n by -1 to 2 do z1 := [op(z1), c(n
        - k + 1, mu)]; tmp := tmp + IncompleteBellB(n, k, op(z1));
        end do;
  return mu*tmp/(1 - mu); end proc;
m := proc(nu, n, mu) local tmp, z, k; option remember; if n = 0 then
      return 1; end if;
  tmp := 0; z := []; for k from n by -1 to 1 do z := [op(z), nu*c(n
        - k + 1, mu)]; tmp := tmp + IncompleteBellB(n, k, op(z));
        end do;
  return tmp; end proc;
```

In particular, the command $\mathrm{c}(2,\mu)$ in Maple yields the second cumulant $\kappa^{(2)}(X) = \mu/(1-\mu)^3$, and by the commands $\mathrm{c}(3,\mu)$ and $\mathrm{c}(4,\mu)$ we find $\kappa^{(3)}(X) = \mu(1+2\mu)/(1-\mu)^5$, and $\kappa^{(4)}(X) = \mu(1+8\mu+6\mu^2)/(1-\mu)^7$, see also (8.85) page 159 of [CF06]. Those results can be recovered from the commands $\mathrm{c}[2,\mu]$, $\mathrm{c}[3,\mu]$ and $\mathrm{c}[4,\mu]$ using the following Mathematica code.

```
c[n_, mu_] := c[n, mu] = (Module[{tmp, k}, If[n == 1, Return[1/(1 -
      mu)]]; tmp = 0; z1 = {};
  For[k = n, k >= 2, k--, z1 = Append[z1, Block[{i = n - k + 1},
       c[i, mu]]]; tmp += BellY[n, k, z1]];
  Simplify[mu*tmp/(1 - mu)]]);
m[nu_ n_, mu_] := (Module[{tmp, z, k}, tmp = 0; If[n == 0,
      Return[1]];
   z = {}; For[k = n, k >= 1, k--, z = Append[z, Block[{i = n - k +
        1}, nu*c[i, mu]]]; tmp += BellY[n, k, z]]; Simplify[tmp]])
```

3 Joint Hawkes cumulants

In the cluster process framework of [HO74], we consider a real-valued self-exciting point process on $[0,\infty)$, with Poisson offspring intensity $\gamma(dx)$ and Poisson immigrant intensity $\nu(dx)$ on $[0,\infty)$, built on the space

$$\Omega = \big\{\xi = \{x_i\}_{i\in I}\subset[0,\infty) \ : \ \#(A\cap\xi)<\infty \text{ for all compact } A\subset[0,\infty)\big\}$$

of locally finite configurations on $[0,\infty)$, whose elements $\xi \in \Omega$ are identified with the Radon point measures $\xi(dz) = \sum_{x\in\xi} \epsilon_x(dz)$, where ϵ_x denotes the Dirac measure at $x \in \mathbb{R}_+$. In particular, any initial immigrant point $y \in \mathbb{R}_+$ branches into a Poisson random sample denoted by $\xi_\gamma(y+dz) = \sum_{x\in\xi} \epsilon_{x+y}(dz)$ and centered at y, with intensity measure $\gamma(y+dz)$ on $[0,\infty)$. Figure 1 presents a graph of the point measure $\xi(dz)$ followed by the corresponding sample paths of the self-exciting counting process $X_t(\xi) := \xi([0,t]) = \sum_{x\in\xi} \mathbf{1}_{[0,t]}(x)$ and its stochastic intensity λ_t, $t \in [0,10]$, in the exponential kernel example of the next section.

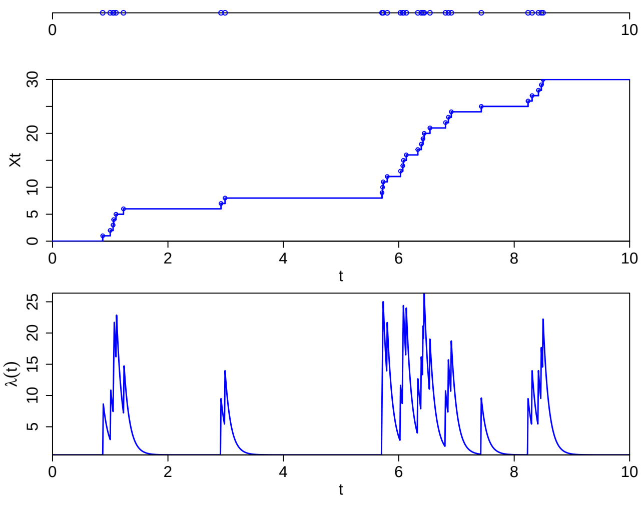

Fig. 1: Sample paths of X_t and of the intensity $\lambda(t)$.

In the sequel, we assume that $\gamma([0,\infty)) < 1$ and consider the integral operator Γ defined as

$$\Gamma f(z) = \int_0^\infty f(z+y)\gamma(dy), \qquad z \in \mathbb{R}_+,$$

and the inverse operator $(I_d - \Gamma)^{-1}$ given by

$$(I_d-\Gamma)^{-1} f(z) = f(z) + \sum_{m=1}^\infty \int_{\mathbb{R}_+^m} f(z+x_1+\cdots+x_m)\gamma(dx_1)\cdots\gamma(dx_m),$$

with

$$(I_d-\Gamma)^{-1}\Gamma f(z) = (I_d-\Gamma)^{-1} f(z) - f(z) = \sum_{m=1}^\infty \int_{\mathbb{R}_+^m} f(z+x_1+\cdots+x_m)\gamma(dx_1)\cdots\gamma(dx_m),$$

$z \in \mathbb{R}_+$. The first cumulant $\kappa_z^{(1)}(f)$ of $\sum_{x\in\xi} f(x)$ given that ξ is started from a single point at $z \in \mathbb{R}_+$ is given by $\kappa_z^{(1)}(f) = (I_d-\Gamma)^{-1} f(z)$ for $n = 1$. The next proposition provides a way to compute the higher order cumulants $\kappa_z^{(n)}(f)$ of $\sum_{x\in\xi} f(x)$ given that ξ is started from a single point at $z \in \mathbb{R}_+$ by an induction relation based on set partitions, see Proposition 3.5 in [Pri21].

Proposition 3.1 *For $n \geq 2$, the joint cumulants $\kappa_z^{(n)}(f_1,\ldots,f_n)$ of $\sum_{x\in\xi} f_1(x),\ldots,\sum_{x\in\xi} f_n(x)$ given that ξ is started from a single point at $z \in \mathbb{R}_+$ are given by the induction relation*

$$\kappa_z^{(n)}(f_1,\ldots,f_n) = \sum_{k=2}^n \sum_{\pi_1\cup\cdots\cup\pi_k=\{1,\ldots,n\}} (I_d-\Gamma)^{-1}\Gamma \prod_{j=1}^k \kappa_z^{(|\pi_j|)}((f_i)_{i\in\pi_j}), \tag{4}$$

$n \geq 2$, where the above sum is over set partitions $\pi_1 \cup \cdots \cup \pi_k = \{1,\ldots,n\}$, $k = 2,\ldots,n$, and $|\pi_i|$ denotes the cardinality of the set $\pi_i \subset \{1,\ldots,n\}$.

The joint cumulants $\kappa^{(n)}(f_1,\ldots,f_n)$ of $\sum_{x\in\xi} f_1(x),\ldots,\sum_{x\in\xi} f_n(x)$ can be obtained as a consequence of Proposition 3.1, by the combinatorial summation

$$\kappa^{(n)}(f_1,\ldots,f_n) = \sum_{k=1}^n \sum_{\pi_1\cup\cdots\cup\pi_k=\{1,\ldots,n\}} \int_0^\infty \prod_{j=1}^k \kappa_z^{(|\pi_j|)}((f_i)_{i\in\pi_j})\nu(dz), \tag{5}$$

see Corollary 3.4 and Proposition 3.5 in [Pri21]. Joint moments can then be recovered by the joint moment-cumulant relation

$$\mathbb{E}\left[\sum_{x\in\xi} f_1(x)\cdots\sum_{x\in\xi} f_n(x)\right] = \sum_{l=1}^n \sum_{\pi_1\cup\cdots\cup\pi_l=\{1,\ldots,n\}} \prod_{j=1}^l \kappa^{(|\pi_j|)}((f_i)_{i\in\pi_j}), \tag{6}$$

which can be inverted as

$$\kappa^{(n)}(f_1,\ldots,f_n) = \sum_{l=1}^n (l-1)!(-1)^{l-1} \sum_{\pi_1\cup\cdots\cup\pi_l=\{1,\ldots,n\}} \prod_{j=1}^l \mathbb{E}\left[\prod_{i\in\pi_j}\sum_{x\in\xi} f_i(x)\right].$$

4 Joint Hawkes moments with exponential kernel

In this section we consider the exponential kernel $\gamma(dx) = a\mathbf{1}_{[0,\infty)}(x)\mathrm{e}^{-bx}dx$, $0 < a < b$, and constant Poisson intensity $\nu(dz) = \nu dz$, $\nu > 0$. In this case,

$$X_t(\xi) := \xi([0,t]) = \sum_{x\in\xi} \mathbf{1}_{[0,t]}(x), \quad t \in \mathbb{R}_+,$$

defines the self-exciting Hawkes process with stochastic intensity

$$\lambda_t := \nu + a\int_0^t \mathrm{e}^{-b(t-s)}dX_s, \qquad t \in \mathbb{R}_+.$$

Here, the integral operator Γ satisfies

$$\Gamma f(z) = a\int_0^\infty f(z+y)\mathrm{e}^{-by}dy, \quad z \in \mathbb{R}_+,$$

and the recursive calculation of joint moments and cumulants will be performed by evaluating $(I_d-\Gamma)^{-1}\Gamma$ in Proposition 3.1 on the family of functions $e_{p,\eta,t}$ of the form $e_{p,\eta,t}(x) := x^p\mathrm{e}^{\eta x}\mathbf{1}_{[0,t]}(x)$, $\eta < b$, $p \geq 0$, as in the next lemma.

Lemma 4.1 *For f in the linear span generated by the functions $e_{p,\eta,t}$, $p \geq 0$, $\eta < b$, the operator $(I_d-\Gamma)^{-1}\Gamma$ is given by*

$$(I_d-\Gamma)^{-1}\Gamma f(z) = a\int_0^{t-z} f(z+y)\mathrm{e}^{(a-b)y}dy,$$

$z \in [0,t]$.

Proof. For all $p, \eta \geq 0$ we have the equality

$$\begin{aligned}
&(I_d-\Gamma)^{-1}\Gamma e_{p,\eta,t}(z)\\
&= \sum_{n=1}^\infty \int_{[0,t]^n} e_{p,\eta,t}(z+x_1+\cdots+x_n)\gamma(dx_1)\cdots\gamma(dx_n)\\
&= \sum_{n=1}^\infty \frac{a^n}{(n-1)!}\int_0^{t-z}(z+y)^p\mathrm{e}^{\eta(z+y)}y^{n-1}\mathrm{e}^{-by}dy\\
&= a\mathrm{e}^{\eta z}\int_0^{t-z}(z+y)^p\mathrm{e}^{(\eta+a-b)y}dy, \quad z\in[0,t],
\end{aligned}$$

which follows from the fact that the sum $\tau_1+\cdots+\tau_n$ of n exponential random variables with parameter $b > 0$ has a gamma distribution with shape parameter $n \geq 1$ and scaling parameter $b > 0$. $\square$

Using Lemma 4.1, we can rewrite (4) for $t_1 < \cdots < t_n$ as

$$\begin{aligned}
&\kappa_z^{(n)}(\mathbf{1}_{[0,t_1]},\ldots,\mathbf{1}_{[0,t_n]}) \qquad (7)\\
&= \sum_{k=2}^n \sum_{\pi_1\cup\cdots\cup\pi_k=\{1,\ldots,n\}} \int_0^{t_1-z} a\mathrm{e}^{(a-b)y}\prod_{j=1}^k \kappa_{z+y}^{(|\pi_j|)}\big((\mathbf{1}_{[0,t_i]})_{i\in\pi_j}\big)dy,
\end{aligned}$$

with

$$\begin{aligned}
\kappa_z^{(1)}(\mathbf{1}_{[0,t]}) &= (I_d-\Gamma)^{-1}\mathbf{1}_{[0,t]}(z)\\
&= \frac{b}{b-a}+\frac{a}{a-b}\mathrm{e}^{(a-b)(t-z)}, \quad z\in[0,t],
\end{aligned}$$

if $a \neq b$, and

$$\kappa_z^{(1)}(\mathbf{1}_{[0,t]}) = (I_d-\Gamma)^{-1}\mathbf{1}_{[0,t]}(z) = 1+a(t-z),$$

$z \in [0,t]$, if $a = b$. The recursive computation of $\kappa_z^{(n)}(\mathbf{1}_{[0,t_1]},\ldots,\mathbf{1}_{[0,t_n]})$ in (7) is implemented in the following Maple and Mathematica codes using Lemma 4.1.

```
kz := proc(z, a, b, t::list) local pm, pp2, p, pp, tmp, k, y, h, i,
       j, ii, u, n, zz, c; option remember; n := nops(t);
  if n = 1 then if a = b then return 1 + a*(t[1] - z); else return
       b/(b - a) + a*exp((a - b)*(t[1] - z))/(a - b); end if; end
       if;
  tmp := 0; pp2 := Iterator:-SetPartitions(n); for pp in pp2 do p :=
       pp2:-ToSets(pp);
  if 2 <= nops(p) then c := 1; for i to nops(p) do c := c*kz(y + z,
       a, b, map(op, convert(p[i], list), t)); end do;
  tmp := tmp + c; end if; end do; return a*int(exp((a - b)*y)*tmp, y
       = 0 .. t[1] - z); end proc;
```

```
Needs["Combinatorica`"];
kz[z_, a_, b_, t__] :=
  kz[z, a, b, t] = (Module[{tmp, y, i, c, n}, n = Length[t];
    If[n == 1, If[a === b, Return[1 + a*(t[[1]] - z)],
      Return[b/(b - a) + a*E^((a - b)*(t[[1]] - z))/(a - b)]]];
    tmp = 0; Do[c = 1; If[Length[p] >= 2, For[i = 1, i <= Length[p],
        i++,
      c = c*Block[{u = y + z, v = t[[p[[i]]]]}, kz[u, a, b, v]]];
      tmp += c], {p, SetPartitions[n]}];
    Return[a*Integrate[E^((a - b)*y)*tmp, {y, 0, t[[1]] - z}]]]);
```

The computation of joint Hawkes cumulants by the recursive relation (5) is then implemented in the following Maple and Mathematica codes.

```
c := proc(a, b, t::list) option remember; local y, e, k, pm, tmp,
      p2, pp, p, c, i, zz, j, u, ii, n; n := nops(t);
  tmp := kz(y, a, b, t); if 2 <= n then pm :=
       Iterator:-SetPartitions(n); for pp in pm do p :=
       pm:-ToSets(pp);
  if 2 <= nops(p) then e := 1; for i to nops(p) do e := e*kz(y, a,
       b, map(op, convert(p[i], list), t)); end do; tmp := tmp + e;
  end if; end do; end if; return int(tmp, y = 0 .. t[1]); end proc;
```

```
c[a_, b_, t__] :=
  c[a, b, t] = (Module[{y, e, tmp, n, i}, n = Length[t]; tmp = 0;
    Do[e = 1; For[i = 1, i <= Length[p], i++,
      e = e*Block[{u = y, v = t[[p[[i]]]]}, kz[u, a, b, v]]];
      tmp += Flatten[{e}][[1]],
      {p, SetPartitions[n]}];
    Return[Integrate[tmp, {y, 0, t[[1]]}]]]);
```

Finally, joint moments are computed from the joint moment-cumulant relation (6) which is implemented in the following Maple and Mathematica codes.

```
m := proc(nu, a, b, t::list) option remember; local y, e, k, u, ii,
      pm, tmp, p2, pp, p, i, zz, j, n; n := nops(t); tmp := nu*c(a,
      b, t);
  if 2 <= n then pm := Iterator:-SetPartitions(n); for pp in pm do p
       := pm:-ToSets(pp); if 2 <= nops(p) then e := 1; for i to
       nops(p) do e := e*nu*c(a, b, map(op, convert(p[i], list),
       t)); end do; tmp := tmp + e;
  end if; end do; end if; return tmp; end proc;
```

```
m[nu_, a_, b_, t__] := (Module[{n, e, i, tmp}, tmp = 0; n =
      Length[t]; If[n == 0, Return[1]];
  Do[e = 1; For[i = 1, i <= Length[p], i++, e = e*nu*c[a, b,
       t[[p[[i]]]]]];
   tmp += e, {p, SetPartitions[n]}]; Flatten[{tmp}][[1]]])
```

The joint moments $\mathbb{E}[X_{t_1}\cdots X_{t_n}]$ of $X_{t_1},\ldots,X_{t_n}$ are obtained from the above code using the command $\mathrm{m}(\nu,a,b,[t_1,\ldots,t_n])$ in Maple or $\mathrm{m}[\nu,a,b,\{t_1,\ldots,t_n\}]$ in Mathematica. Figures 2 to 7 are plotted with $\nu=1$, $a=0.5$, $b=1$, $T=2$, and one million Monte Carlo samples, and Figure 2 presents the first moment $m_1(t)=\mathrm{m}(\nu,a,b,[t])=\mathrm{m}[\nu,a,b,\{t\}]$.

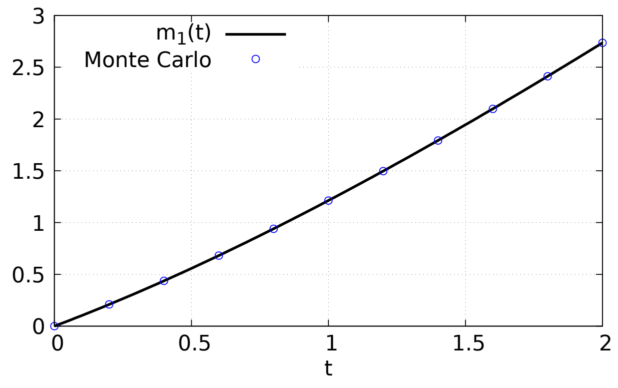

Fig. 2: First moment.

For example, the second joint moment of (X_t, X_T) is obtained by the command $m_2(t,T) = \mathrm{m}(\nu, a, b, [t,T])$ in Maple or $m_2(t,T) = \mathrm{m}[\nu, a, b, \{t,T\}]$ in Mathematica, which yields

$$\begin{aligned}&\mathbb{E}[X_t X_T] = \\ &\frac{\nu}{(a-b)^4}\Bigg(\frac{1}{2}\mathrm{e}^{-at-b(t+T)}(2b^4 t\mathrm{e}^{at+b(t+T)} \\ &+ a^2 b(-\mathrm{e}^{a(2t+T)} + \mathrm{e}^{2bt+aT} + 2bt\mathrm{e}^{2at+bT} \\ &+ 2bt\mathrm{e}^{bt+a(t+T)}) + 2a^3(\mathrm{e}^{a(2t+T)} - bt\mathrm{e}^{2at+bT} \\ &- \mathrm{e}^{bt+a(t+T)}(1+bt)) - 2ab^2(\mathrm{e}^{2bt+aT} - 2\mathrm{e}^{2at+bT} \\ &- \mathrm{e}^{bt+a(t+T)} + \mathrm{e}^{at+b(t+T)}(2+bt))) \\ &+ \nu(b^2 t + a(\mathrm{e}^{(a-b)t} - bt - 1)) \\ &\quad\times(b^2 T + a(\mathrm{e}^{(a-b)T} - bT - 1))\Bigg),\end{aligned}$$

see Figure 3.

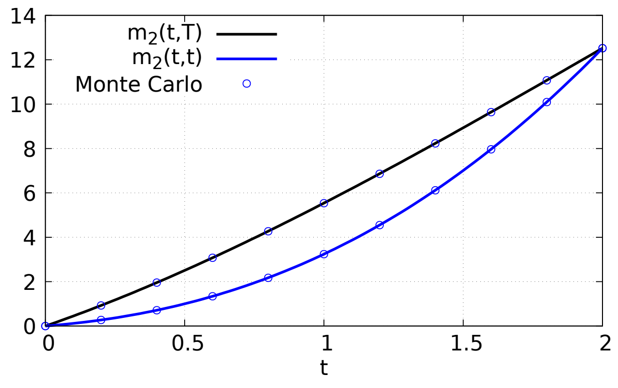

Fig. 3: Second joint moment.

Figures 4 and 5 show the numerical evaluation of third and fourth joint moments, obtained from $m_3(t_1,t_2,t_3) = \mathrm{m}(\nu,a,b,[t_1,t_2,t_3]) = \mathrm{m}[\nu,a,b,\{t_1,t_2,t_3\}]$ and $m_4(t_1,t_2,t_3,t_4) = \mathrm{m}(\nu,a,b,[t_1,t_2,t_3,t_4]) = \mathrm{m}[\nu,a,b,\{t_1,t_2,t_3,t_4\}]$.

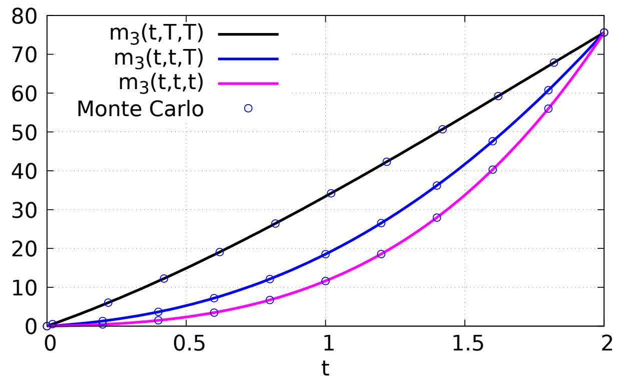

Fig. 4: Third joint moments.

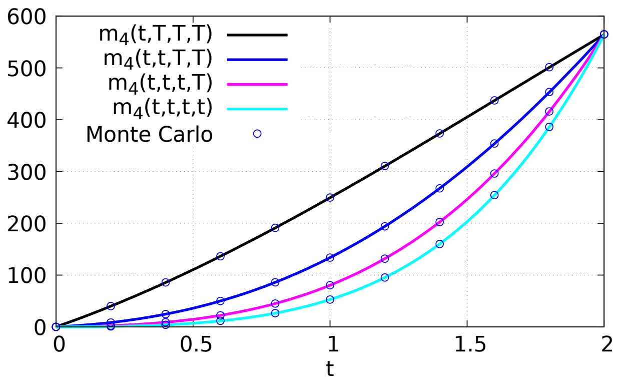

Fig. 5: Fourth joint moments.

The following tables count the (approximate) numbers of summands appearing in joint cumulant and moment expressions when expanded as a sum of the form

$$\sum_{\substack{k,l,p_1,\ldots,p_n\geq 0\\ q_1,\ldots,q_n,r_1,\ldots,r_n\geq 0}} a^k b^l t_1^{p_1}\cdots t_n^{p_n}\mathrm{e}^{q_1 a t_1+\cdots+q_n a t_n + r_1 b t_1+\cdots+r_n b t_n},$$

excluding factorizations and simplifications of expressions.

	One variable			
Cumulant	Time	Count	All variables	
Sixth	64s	671	Time	Count
Fifth	11s	226	2403s	3288
Fourth	1.7s	81	31s	536
Third	0.5s	35	1.6s	91
Second	0.2s	12	0.3s	14
First	0.05s	4		

Table 1: Counts of summands and cumulant computation times in Maple.

The tables also display the corresponding computation times on a 8-core laptop computer with 8Gb RAM. Symbolic computation appears faster with Maple, although computation times become similar at the order six and above.
Figures 6 and 7 show the numerical evaluation of fifth and sixth joint moments. $m_5(t_1,t_2,t_3,t_4,t_5)$ and $m_6(t_1,t_2,t_3,t_4,t_5,t_6)$.

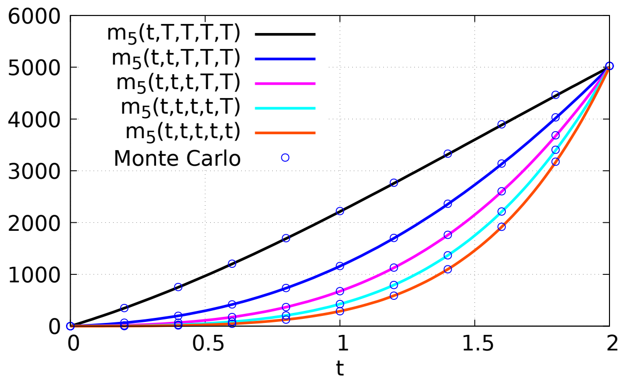

Fig. 6: Fifth joint moments.

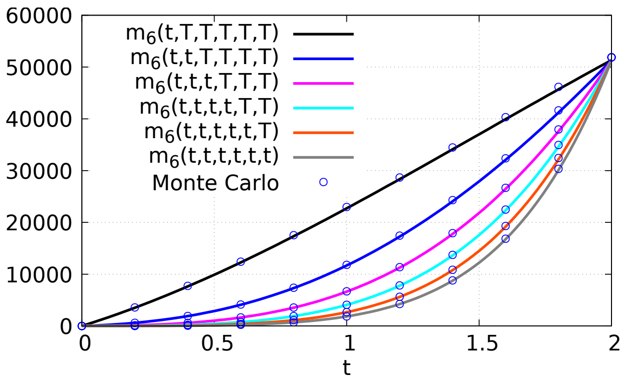

Fig. 7: Sixth joint moments.

Computation times are presented in seconds for symbolic calculation using generic variables $a, b, t_1, \ldots, t_n$, and can be significantly shorter when the variables are set to specific numerical values. Moment computations times in Table 2 are similar to those of Table 1, and can be sped up if cumulant functions are memoized after repeated calls.

	One variable			
Moment	Time	Count	All variables	
Sixth	66s	2159	Time	Count
Fifth	11s	762	2544s	27116
Fourth	1.9s	265	35s	2236
Third	0.5s	88	1.8s	266
Second	0.2s	22	0.5s	29
First	0.06s	4		

Table 2: Counts of summands and moment computation times in Maple.

One-variable examples are computed using the following Maple code which uses Bell polynomials instead of sums over partitions.

```
kz := proc(z, n, a, b, t) local tmp, z1, y, k; option remember; if n
      = 1 then if a = b then return 1 + a*(t - z); else return b/(b
      - a) + a*exp((a - b)*(t - z))/(a - b); end if; end if; tmp :=
      0; z1 := []; for k from n by -1 to 2 do z1 := [op(z1), kz(y +
      z, n - k + 1, a, b, t)]; tmp := tmp + IncompleteBellB(n, k,
      op(z1)); end do; return a*int(exp((a - b)*y)*tmp, y = 0 .. t -
      z); end proc;
```

```
c := proc(a, b, t, n) local y, k, z; option remember; y := []; for k
      from n by -1 to 1 do y := [op(y), kz(z, n - k + 1, a, b, t)];
      end do; return int(CompleteBellB(n, op(y)), z = 0 .. t); end
      proc;
m:=proc(nu,a,b,t,n) local z,k;option remember;if n=0 then return 1
      end if;z:=[];for k from n by -1 to 1 do
      z:=[op(z),nu*c(a,b,t,n-k+1)]; end do; return
      CompleteBellB(n,op(z));end proc;
```

For example, the second moment of X_t is obtained in Maple by the command $\mathrm{m}(\nu, a, b, t, 2)$, which yields

$$\begin{aligned}\mathbb{E}[X_t^2] =&\nu^2 \frac{(b^2 t + a(-1 + \mathrm{e}^{(a-b)t} - bt))^2}{(a-b)^4} \\ &+ \frac{\nu}{2(a-b)^4} \big(\mathrm{e}^{-2bt}(2b^4 \mathrm{e}^{2bt} t \\ &+ a^2 b(-\mathrm{e}^{2at} + \mathrm{e}^{2bt} + 4b\mathrm{e}^{(a+b)t} t) \\ &- 2ab^2(-3\mathrm{e}^{(a+b)t} + \mathrm{e}^{2bt}(3 + bt)) \\ &+ 2a^3(\mathrm{e}^{2at} - \mathrm{e}^{(a+b)t}(1 + 2bt)))\big).\end{aligned}$$

The same result can be obtained in Mathematica from the command $\mathrm{m}[\nu, a, b, t, 2]$ using the code below.

```
kz[z_, n_, a_, b_, t_] :=
  kz[z, n, a, b, t] = (Module[{tmp, k, y, z1},
    If[n == 1, If[a === b, Return[1 + a*(t - z)],
      Return[b/(b - a) + a*E^((a - b)*(t - z))/(a - b)]]]; tmp = 0;
    z1 = {}; For[k = n, k >= 2, k--, z1 = Append[z1, Block[{i = n -
          k + 1, u = y + z}, kz[u, i, a, b, t]]];
     tmp += BellY[n, k, z1];]; a*Integrate[E^((a - b)*y)*tmp, {y, 0,
          t - z}]]);
c[a_, b_, t_, n_] := c[a, b, t, n] = (Module[{y, k, z, temp}, temp =
      0; y = {}; For[k = n, k >= 1, k--,
     y = Append[y, Block[{i = n - k + 1, u = z}, kz[u, i, a, b,
          t]]]; temp += BellY[n, k, y]]; Return[Integrate[temp,
          {z, 0, t}]]]);
m[nu_, a_, b_, t_, n_] := m[a, b, t, n] = (Module[{tmp, z, k}, tmp =
      0; If[n == 0, Return[1]]; z = {}; For[k = n, k >= 1, k--, z =
      Append[z, nu*c[a, b, t, n - k + 1]]; tmp += BellY[n, k, z]];
      tmp])
```

5 Joint intensity moments with exponential kernel

More generally, Equations (4) and (5) can be implemented for any family $(f_1, \ldots, f_n)$ of functions defined on $\mathbb{R}_+$ by modifying the above Maple and Mathematica codes as follows.

```
kz := proc(z, a, b, f::list, t::list) local pm, pp2, p, pp, tmp, k,
      y, h, i, j, ii, u, n, zz, c; option remember; n := nops(t); if
      n = 1 then return f[1](z, a, b, t[1]) + a*int(exp((a -
      b)*y)*f[1](z + y, a, b, t[1]), y = 0 .. t[1] - z); end if; tmp
      := 0; pp2 := Iterator:-SetPartitions(n);
  for pp in pp2 do p := pp2:-ToSets(pp); if 2 <= nops(p) then c :=
       1; for i to nops(p) do c := c*kz(z + y, a, b, map(op,
       convert(p[i], list), f), map(op, convert(p[i], list), t));
       end do; tmp := tmp + c; end if; end do;
  return a*int(exp((a - b)*y)*tmp, y = 0 .. t[1] - z); end proc
c := proc(a, b, f::list, t::list) local y, e, k, pm, tmp, p2, pp, p,
      c, i, zz, j, u, ii, n; option remember; n := nops(t); tmp :=
      kz(y, a, b, f, t); if 2 <= n then pm :=
      Iterator:-SetPartitions(n);
  for pp in pm do p := pm:-ToSets(pp); if 2 <= nops(p) then e := 1;
       for i to nops(p) do e := e*kz(y, a, b, map(op, convert(p[i],
       list), f), map(op, convert(p[i], list), t)); end do; tmp :=
       tmp + e; end if; end do; end if;
  return int(tmp, y = 0 .. t[1]); end proc
m := proc(nu, a, b, f::list, t::list) local y, e, k, u, ii, pm, tmp,
      p2, pp, p, i, zz, j, n; option remember; n := nops(t); tmp :=
      nu*c(a, b, f, t); if 2 <= n then pm :=
      Iterator:-SetPartitions(n);
  for pp in pm do p := pm:-ToSets(pp); if 2 <= nops(p) then e := 1;
       for i to nops(p) do e := e*nu*c(a, b, map(op, convert(p[i],
       list), f), map(op, convert(p[i], list), t)); end do; tmp :=
       tmp + e; end if; end do; end if;
  return tmp; end proc
```

For example, defining f:=(x,a,b,t)→a∗exp(-b∗(t-x)), the command m(ν,a,b,[1,f],[t,t]) in Maple, or m[ν,a,b,{1&,f},{t,t}] in Mathematica using the code below, yields the joint moment

$$
\begin{aligned}
&a\mathbb{E}\left[N_t\int_0^t \mathrm{e}^{-b(t-s)}dN_s\right] \\
&=\frac{\nu a}{2(a-b)^3}\Big((a-2b)b+a(2a-b)\mathrm{e}^{2(a-b)t} \\
&\quad-2(a-b)\mathrm{e}^{(a-b)t}(a+b+abt) \\
&\quad+2\nu\big(\mathrm{e}^{(a-b)t}-1\big)\big(b^2t+a(-1+\mathrm{e}^{(a-b)t}-bt)\big)\Big).
\end{aligned}
$$

```
Needs["Combinatorica`"]
kz[z_, a_, b_, f__, t__] := kz[z, a, b, f, t] = (Module[{tmp, y, i,
       c, n, u, v, g}, n = Length[t];
     If[n == 1,
      Return[f[[1]][z, a, b, t[[1]]] +
        a*Integrate[
          f[[1]][z + y, a, b, t[[1]]]*E^((a - b)*y), {y, 0,
           t[[1]] - z}]]]; tmp = 0;
     Do[c = 1;
      If[Length[p] >= 2,
       For[i = 1, i <= Length[p], i++,
        c = c*Block[{u = y + z, g = f[[p[[i]]]], v = t[[p[[i]]]]},
           kz[u, a, b, g, v]]]; tmp += c], {p, SetPartitions[n]}];
     Return[a*Integrate[E^((a - b)*y)*tmp, {y, 0, t[[1]] - z}]]]);
c[a_, b_, f__, t__] := c[a, b, f, t] = (Module[{y, e, tmp, n, i, u,
       v, g}, n = Length[t]; tmp = 0;
     Do[e = 1;
      For[i = 1, i <= Length[p], i++,
       e = e*Block[{u = y, g = f[[p[[i]]]], v = t[[p[[i]]]]},
          kz[u, a, b, g, v]]];
      tmp += Flatten[{e}][[1]], {p, SetPartitions[n]}];
     Return[Integrate[tmp, {y, 0, t[[1]]}]]]);
m[nu_, a_, b_, f__, t__] := m[nu, a, b, f, t] = (Module[{n, e, i,
       tmp}, tmp = 0;
    n = Length[t]; If[n == 0, Return[1]];
    Do[e = 1; For[i = 1, i <= Length[p], i++,
                e = nu*e*c[a, b, f[[p[[i]]]], t[[p[[i]]]]]];
     tmp += e, {p, SetPartitions[n]}]; Flatten[{tmp}][[1]]])
```

Similarly, the command
ν^2 + 2 ν m(ν,a,b,[f],[t]) + m(ν,a,b,[f,f],[t,t])
in Maple or
ν^2 + 2ν m[ν,a,b,{f},{t}) + m[ν,a,b,{f,f},{t,t})
in Mathematica yields the intensity second moment

$$
\begin{aligned}
&\mathbb{E}\left[\left(\nu+a\int_0^t \mathrm{e}^{-b(t-s)}dN_s\right)^2\right] \\
&=\nu^2\mathrm{e}^{2(a-b)t} \\
&+\nu(2\nu b+a^2)\left(b\frac{1-\mathrm{e}^{2(a-b)t}}{2(a-b)^2}-a\frac{\mathrm{e}^{(a-b)t}-\mathrm{e}^{2(a-b)t}}{(a-b)^2}\right),
\end{aligned}
$$

cf., e.g., Corollary 3 in [BSS18].

References

[ABG+17] M. Achab, E. Bacry, S. Gaïffas, I. Mastromatteo, and J.-F. Muzy. Uncovering causality from multivariate Hawkes integrated cumulants. *Journal of Machine Learning Research*, 18(1):6998–7025, 2017.

[ABMR18] M. Achab, E. Bacry, J. F. Muzy, and M. Rambaldi. Analysis of order book flows using a non-parametric estimation of the branching ratio matrix. *Quant. Finance*, 18(2):199–212, 2018.

[BDM12] E. Bacry, K. Dayri, and J.F. Muzy. Non-parametric kernel estimation for symmetric Hawkes processes. Application to high frequency financial data. *Eur. Phys. J. B*, 85:157–168, 2012.

[BSS18] G. Bernis, K. Salhi, and S. Scotti. Sensitivity analysis for marked Hawkes processes: application to CLO pricing. *Math. Financ. Econ.*, 12(4):541–559, 2018.

[CF06] P.C. Consul and F. Famoye. *Lagrangian probability distributions*. Birkhäuser Boston, Inc., Boston, MA, 2006.

[CHY20] L. Cui, A. Hawkes, and H. Yi. An elementary derivation of moments of Hawkes processes. *Adv. in Appl. Probab.*, 52:102–137, 2020.

[CR10] S. Cardanobile and S. Rotter. Multiplicatively interacting point processes and applications to neural modeling. *Journal of Computational Neuroscience*, 28:267–284, 2010.

[DP20] A. Daw and J. Pender. Matrix calculations for moments of Markov processes. Preprint arXiv:1909.03320, 2020.

[DZ11] A. Dassios and H. Zhao. A dynamic contagion process. *Adv. in Appl. Probab.*, 43:814–846, 2011.

[EGG10] E. Errais, K. Giesecke, and L.R. Goldberg. Affine point processes and portfolio credit risk. *SIAM Journal on Financial Mathematics*, 1:642–665, 2010.

[ELL11] P. Embrechts, T. Liniger, and L. Lin. Multivariate Hawkes processes: an application to financial data. *J. Appl. Probab.*, 48:367–387, 2011.

[Haw71] A.G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58:83–90, 1971.

[HO74] A.G. Hawkes and D. Oakes. A cluster process representation of a self-exciting process. *J. Appl. Probab.*, 11(3):493–503, 1974.

[JHR15] S. Jovanović, J. Hertz, and S. Rotter. Cumulants of Hawkes point processes. *Phys. Rev. E*, 91, 2015.

[LRV] H. Lacoin, R. Rhodes, and V. Vargas. A probabilistic approach of ultraviolet renormalisation in the boundary Sine-Gordon model. Preprint arXiv:1903.01394, 28 pages, 2020, to appear in Probability Theory and Related Fields.

[LS59] V.P. Leonov and A.N. Shiryaev. On a method of calculation of semi-invariants. *Theory Probab. Appl.*, 4:319–329, 1959.

[Luk55] E. Lukacs. Applications of Faà di Bruno's formula in mathematical statistics. *Amer. Math. Monthly*, 62:340–348, 1955.

[McC87] P. McCullagh. *Tensor methods in statistics*. Monographs on Statistics and Applied Probability. Chapman & Hall, London, 1987.

[MMG20] L. Montangie, C. Miehl, and J. Gjorgjieva. Autonomous emergence of connectivity assemblies via spike triplet interactions. *PLoS Comput Biol*, 16(5):1–44, 2020.

[MSB$^+$11] G.O. Mohler, M.B. Short, P.J. Brantingham, F.P. Schoenberg, and G.E. Tita. Self-exciting point process modeling of crime. *J. Amer. Statist. Assoc.*, 106(493):100–108, 2011.

[Oga98] Y. Ogata. Space-time point-process models for earthquake occurrences. *Ann. Inst. Statist. Math.*, 50(2):379–402, 1998.

[OJSBB17] G.K. Ocker, K. Josić, E. Shea-Brown, and M.A. Buice. Linking structure and activity in nonlinear spiking networks. *PLoS Comput Biol*, 16(3):1–47, 2017.

[Pri21] N. Privault. Recursive computation of the Hawkes cumulants. *Statist. Probab. Lett.*, 177:Article 109161, 2021.

[PS98] G. Pólya and G. Szegö. *Problems and Theorems in Analysis I.* Springer, 1998. Reprint of the 1978 Edition.

[RBS10] P. Reynaud-Bouret and S. Schbath. Adaptive estimation for Hawkes processes; application to genome analysis. *Ann. Statist.*, 38(5):2781–2822, 2010.

[RLMX18] M.-A. Rizoiu, Y. Lee, S. Mishra, and L. Xie. Hawkes processes for events in social media. In Shih-Fu Chang, editor, *Frontiers of multimedia research*, volume 17 of *ACM Books*, pages 230–262. Association for Computing Machinery and Morgan & Claypool Publishers, 2018.

[SHH19] F. P. Schoenberg, M. Hoffmann, and R.J. Harrigan. A recursive point process model for infectious diseases. *Ann. Inst. Statist. Math.*, 71:1271–1287, 2019.

[Sta99] R.P. Stanley. *Enumerative combinatorics. Vol. 2*, volume 62 of *Cambridge Studies in Advanced Mathematics.* Cambridge University Press, Cambridge, 1999.

[YSB21] B. Yuan, F.P. Schoenberg, and A.L. Bertozzi. Fast estimation of multivariate spatiotemporal Hawkes processes and network reconstruction. *Ann. Inst. Statist. Math.*, 73:1127–1152, 2021.