OPERATIONS RESEARCH
Vol. 00, No. 0, Xxxxx 0000, pp. 000–000
ISSN 0030-364X | EISSN 1526-5463 | 00 | 0000 | 0001

INFORMS
DOI 10.1287/xxxx.0000.0000
© 0000 INFORMS

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

Rate-Based Daily Arrival Process Models with Application to Call Centers

Boris N. Oreshkin, Nazim Régnard, Pierre L'Ecuyer
Département d'Informatique et de Recherche Opérationnelle, Pavillon Aisenstadt, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, Québec, Canada H3C 3J7, lecuyer@iro.umontreal.ca

---

Online Supplement

In this Online Supplement, we detail the methods we have developed to compute (or approximate) MLEs for the PG2, PG2pow, and PGnorta models. Section A deals with the PG2 model. In Section A.1, we derive MMEs for that model. These estimators turn out to be less reliable and robust than the MLEs, but we use them as starting points in our optimization algorithm to compute the MLEs. In Section A.2, we derive an expression for the log-likelihood function and find that it has an integral form that we do not know how to evaluate exactly. In Section A.3, we develop a Monte-Carlo estimator of this log-likelihood, in functional form. This provides a sample average approximation (SAA) of the exact likelihood function. We show that this SAA converges uniformly to the exact one, and that its optimal value and set of optimizers converge as well, when the Monte Carlo sample size increases. For a fixed Monte Carlo sample, this SAA is in fact a smooth deterministic function which can be optimized by known nonlinear optimization techniques. In Section A.4 we derive expressions for the derivatives of this SAA with respect to each parameter. These derivatives can be used in a nonlinear optimization algorithm that maximizes the SAA, as discussed in Section A.5. This approximate optimizer of the SAA can be taken as an approximate optimizer of the exact likelihood function. In Section A.6, we extend this methodology to cover a restricted form of the PG2 model where the shape parameter α_j of the busyness factor for period

j must follow a smooth function of j. In Sections B and C, we explain how we compute the MLEs for the PG2pow and PGnorta models. In Section D, we explain the kernel density estimators we have used to compute 95% confidence intervals. In Section E, we report some additional correlation plots for our data.

In this Supplement, we suppose that we have data for I (independent) days of operation of the call center. We denote by $\mathbf{X}_i = (X_{i,1}, \ldots, X_{i,p})$ the vector of arrival counts for day i, and let $\mathbb{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_I)$.

Appendix A: Parameter Estimation for the Two-Level Busyness Factor Model PG2

A.1. Moment Matching Estimators

The easiest way of estimating the parameters of the PG2 model is via moment matching. It only provides a crude estimator that is not always very reliable, but it is simple and easy to compute for all model parameters, and can be used very conveniently as a starting point in the optimization algorithm for MLEs.

The empirical means, variances, and covariances of the $X_{i,j}$'s are:

$$\widehat{\mu}_j = \frac{1}{I}\sum_{i=1}^{I} X_{i,j},$$
$$\widehat{\sigma}_j^2 = \frac{1}{I}\sum_{i=1}^{I} (X_{i,j} - \widehat{\mu}_j)^2,$$
$$\widehat{r}_{j,k} = \frac{1}{I}\sum_{i=1}^{I} (X_{i,j} - \widehat{\mu}_j)(X_{i,k} - \widehat{\mu}_k)$$

We match the moments of our model to the empirical moments separately for the means, variances, and covariances.

Matching the means gives the MME for the base rates: $\widehat{\lambda}_j = \widehat{\mu}_j$. Given this estimate, by matching the covariances, solving the least squares matching problem, and after some algebraic manipulations, we obtain the estimator of β:

$$\widehat{\beta} = \arg\min_{\beta} \sum_{j=1}^{p-1}\sum_{k=j+1}^{p} \left(\widehat{r}_{j,k} - \frac{\lambda_j\lambda_k}{\beta}\right)^2 = \frac{\sum_{j=1}^{p-1}\sum_{k=j+1}^{p}\widehat{\mu}_j\widehat{\mu}_k}{\sum_{j=1}^{p-1}\sum_{k=j+1}^{p}\widehat{r}_{j,k}}. \quad (1)$$

On close examination, we see that this estimator can take a negative value if the sum of empirical covariances in (1) is negative. When this happens, it means that the sum of covariances is likely to be close to 0, and therefore we suggest to replace $\hat{\beta}$ by ∞ (or a very large value), which means that $\bar{B}$ has a degenerate distribution with mean 1 and variance 0. This is equivalent to removing $\bar{B}$ from the model, and brings us back to the PGindep model.

Finally, by matching the variance we obtain the estimator of α_j:

$$\widehat{\alpha}_j = \arg\min_{\alpha_j} \left(\widehat{\sigma}_j^2 - \lambda_j - \frac{(1+\beta+\alpha_j)\lambda_j^2}{\beta\alpha_j} \right)^2 = \frac{(1+\widehat{\beta})\widehat{\mu}_j^2}{\widehat{\beta}\widehat{\sigma}_j^2 - \widehat{\mu}_j^2 - \widehat{\beta}\widehat{\mu}_j}. \tag{2}$$

Here too it could happen that this estimator takes a negative value, if the empirical variance $\widehat{\sigma}_j^2$ is too small. In this case, we remove $\tilde{B}_j$ from the model, or assume (equivalently) that $\mathrm{Var}(\tilde{B}_j) = 0$.

In our experiments, we found that this estimator of α_j is sometimes very noisy, because it uses noisy single period estimates $\widehat{\sigma}_j^2$ and $\widehat{\mu}_j$ (in contrast to $\hat{\beta}$ which uses averages). Problems may arise especially when the denominator is close to zero. This difficulty can be alleviated to some extent via smoothing (akin to the moving-average filter), by replacing (2) by

$$\widehat{\alpha}_j = \arg\min_{\alpha_j} \sum_{k=j-M}^{j+M} \left(\widehat{\sigma}_k^2 - \lambda_j - \frac{(1+\beta+\alpha_j)\lambda_j^2}{\beta\alpha_j} \right)^2.$$

Here, $2M+1$ is the size of window within which we assume α_j to be constant. In this case, we obtain a more robust and often better behaved estimator:

$$\widehat{\alpha}_j = (1+\widehat{\beta}) \left(\widehat{\beta} \frac{\sum_{k=j-M}^{j+M} \widehat{\mu}_k^2 (\widehat{\sigma}_k^2 - \widehat{\mu}_k)}{\sum_{k=j-M}^{j+M} \widehat{\mu}_k^4} - 1 \right)^{-1}.$$

In our experiments, we had good results when using this estimator with $M = 5$. Still, the MLE developed in forthcoming subsections is generally better than this modified MME.

A.2. The Likelihood and Log-Likelihood Functions

For the PG2 model, let $\tilde{B}_{i,j}$ be the realization of the busyness factor $\tilde{B}_j$ of period j and $\bar{B}_i$ the realization of the daily busyness factor $\bar{B}$, for day i of the data set. All these busyness factors are unobserved (unknown). We denote the vector of daily busyness factors by $\mathbb{B} = (\bar{B}_1, \ldots, \bar{B}_I)$ and define $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)$. In our model, the counts are conditionally independent given the realizations of the busyness factors, so the joint distribution of the (observed) counts conditional on $\mathbb{B}$ can be written as

$$\begin{aligned} p(\mathbb{X}|\mathbb{B}, \beta, \boldsymbol{\alpha}, \boldsymbol{\lambda}) &= \int_0^\infty \cdots \int_0^\infty \prod_{i=1}^{I} \prod_{j=1}^{p} \frac{(\lambda_j \tilde{B}_{i,j} \bar{B}_i)^{X_{i,j}} e^{-\lambda_j \tilde{B}_{i,j} \bar{B}_i}}{X_{i,j}!} \frac{\alpha_j^{\alpha_j} \tilde{B}_{i,j}^{\alpha_j - 1} e^{-\alpha_j \tilde{B}_{i,j}}}{\Gamma(\alpha_j)} d\tilde{B}_{i,j} \\ &= \prod_{i=1}^{I} \prod_{j=1}^{p} \int_0^\infty \frac{(\lambda_j \tilde{B}_{i,j} \bar{B}_i)^{X_{i,j}} e^{-\lambda_j \tilde{B}_{i,j} \bar{B}_i}}{X_{i,j}!} \frac{{\alpha_j}^{\alpha_j} \tilde{B}_{i,j}^{\alpha_j - 1} e^{-\alpha_j \tilde{B}_{i,j}}}{\Gamma(\alpha_j)} d\tilde{B}_{i,j} \\ &= \left[\prod_{j=1}^{p} \frac{{\alpha_j}^{I\alpha_j}}{\Gamma(\alpha_j)^I} \right] \prod_{i=1}^{I} \prod_{j=1}^{p} \frac{\Gamma(\alpha_j + X_{i,j})}{X_{i,j}!} \frac{(\bar{B}_i \lambda_j)^{X_{i,j}}}{(\alpha_j + \bar{B}_i \lambda_j)^{X_{i,j} + \alpha_j}}. \end{aligned}$$

Note that we are conditioning on the daily busyness factors $\bar{B}_i$, but integrating with respect to the $\tilde{B}_{i,j}$'s. Then, we can write the likelihood function by integrating out with respect to the $\bar{B}_i$:

$$p(\mathbb{X}|\beta, \boldsymbol{\alpha}, \boldsymbol{\lambda}) = \left[\prod_{j=1}^{p} \frac{{\alpha_j}^{I\alpha_j}}{\Gamma(\alpha_j)^I} \right] \left[\prod_{i=1}^{I} \prod_{j=1}^{p} \frac{\Gamma(\alpha_j + X_{i,j})}{X_{i,j}!} \right] \prod_{i=1}^{I} \int_0^\infty \left[\prod_{j=1}^{p} \frac{(\bar{B}_i \lambda_j)^{X_{i,j}}}{(\alpha_j + \bar{B}_i \lambda_j)^{X_{i,j} + \alpha_j}} \right] \frac{\beta^\beta \bar{B}_i^{\beta-1} e^{-\bar{B}_i \beta}}{\Gamma(\beta)} d\bar{B}_i. \tag{3}$$

For convenience, we will work with the log-likelihood function

$$L = L(\beta, \boldsymbol{\alpha}, \boldsymbol{\lambda}) = \log p(\mathbb{X}|\beta, \boldsymbol{\alpha}, \boldsymbol{\lambda}), \tag{4}$$

which is the logarithm (in natural basis) of (3) and has the same maximizer. Using the shorthand notation

$$\varphi(\boldsymbol{\alpha}, \mathbb{X}) = \left[\prod_{j=1}^{p} {\alpha_j}^{I\alpha_j}/\Gamma(\alpha_j)^I\right] \prod_{i=1}^{I}\prod_{j=1}^{p} \frac{\Gamma(\alpha_j + X_{i,j})}{X_{i,j}!}, \tag{5}$$

$$\Theta_i(\bar{B}_i, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) = \prod_{j=1}^{p} \frac{(\bar{B}_i\lambda_j)^{X_{i,j}}}{(\alpha_j + \bar{B}_i\lambda_j)^{X_{i,j}+\alpha_j}}, \tag{6}$$

$$p(\bar{B}_i|\beta) = \frac{\beta^\beta \bar{B}_i^{\beta-1} e^{-\bar{B}_i\beta}}{\Gamma(\beta)}, \tag{7}$$

we can rewrite

$$L = \log\varphi(\boldsymbol{\alpha}, \mathbb{X}) + \sum_{i=1}^{I} \log \int_0^\infty \Theta_i(\bar{B}_i, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) p(\bar{B}_i|\beta) d\bar{B}_i. \tag{8}$$

To avoid numerical problems and to ensure convergence when optimizing the log-likelihood, we will assume that the parameter $\boldsymbol{\theta} = (\beta, \boldsymbol{\alpha}, \boldsymbol{\lambda})$ lies in the compact set $\mathcal{S} = [\epsilon, K_1]^{p+1} \times [0, K_2]^p$, for some constants $\epsilon > 0$, $K_1 > \epsilon$, and $K_2 > 0$, and restrict our maximization to that compact set. Reasonable values for these constants can be selected after a quick assessment of the data. One can easily verify that given the data, the log-likelihood function L is continuous and bounded in $\mathcal{S}$, so it has at least one maximizer $\boldsymbol{\theta}_0 \in \mathcal{S}$. We have no proof that there cannot be multiple local maximizers, but we never had a problem with this in our experiments, either for L or its estimator $\hat{L}_N$ defined in the next subsection.

A.3. Estimating the Log-likelihood Function

One significant problem in maximizing L is that we do not know how to evaluate exactly the integrals with respect to $\bar{B}_i$ in (8). We will replace these integrals, as functions of the parameter vector $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \beta, \boldsymbol{\lambda})$, by Monte Carlo estimates, and then optimize the resulting estimated log-likelihood function. This approach is known as the sample average approximation (SAA) method Ruszczyński and Shapiro (2003). SAA replaces the log-likelihood function L by an estimated log-likelihood function $\hat{L}_N$, defined by estimating all the integrals with a fixed sample size N and with common random numbers across all values of $\boldsymbol{\theta}$, This sample function $\hat{L}_N$ (which becomes deterministic) is then optimized via any nonlinear optimization algorithm. Convergence of SAA is usually studied in two steps: (a) proving that the approximate log-likelihood function $\hat{L}_N$ converges to L in a sufficiently strong way when $N \to \infty$, so that its optimal value and its set of optimizers

converge to those of L in some sense; and (b) showing that the retained optimization algorithm converges to an optimum of $\hat{L}_N$. We start by defining $\hat{L}_N$ more precisely and addressing (a).

To construct $\hat{L}_N$, we proceed as follows. For each i, we generate an i.i.d. sample $\xi_i^{(1)} = \xi_i^{(1)}(\beta), \ldots, \xi_i^{(N)} = \xi_i^{(N)}(\beta)$ from the gamma density $p(\bar{B}_i|\beta)$ and define the "weights"

$$\omega_i^{(n)} = \omega_i^{(n)}(\beta) = \Theta_i(\xi_i^{(n)}(\beta), \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}), \quad n = 1, \ldots, N. \tag{9}$$

The log-likelihood function estimator is defined using these sample weights:

$$\hat{L}_N = \hat{L}_N(\beta, \boldsymbol{\alpha}, \boldsymbol{\lambda}) = \log \varphi(\boldsymbol{\alpha}, \mathbb{X}) + \sum_{i=1}^{I} \log \left[\frac{1}{N} \sum_{n=1}^{N} \omega_i^{(n)}(\beta) \right]. \tag{10}$$

We further assume that the samples $\xi_i^{(n)}(\beta)$ are drawn by inversion of the gamma cdf, with common random numbers across all values of β. That is, we generate $u_i^{(1)}, \ldots, u_i^{(N)}$ from the uniform distribution over $(0,1)$, and put $\xi_i^{(n)}(\beta) = G_\beta^{-1}(u_i^{(n)})$, where $G_\beta^{-1}(\cdot)$ is the inverse of $G_\beta(\cdot)$, the gamma cdf with mean 1 and variance $1/\beta$. Note that $\hat{L}_N$ is defined as a function of the parameters, for fixed realizations of these uniform random variables.

With the standard (or crude) Monte Carlo approach, the random numbers $u_i^{(n)}$ are taken as independent. One may also improve the accuracy (reduce the variance without introducing bias) by introducing some dependence between those random numbers. One way to do this is by taking a stratified sample over the interval $(0,1)$: draw $u_i^{(1)}$ uniformly from the interval $(0, 1/N)$, $u_i^{(2)}$ uniformly from the interval $(1/N, 2/N)$, etc. Another way is to take a randomized quasi-Monte Carlo (RQMC) point set for these random numbers (L'Ecuyer 2009). As an example, one may take a randomly-shifted lattice followed by a baker's transformation, defined as follows: generate a single uniform random number u over $(0,1)$, put $u_i^{(n)} = 2(n-1+u)/N$ for $n = 1, \ldots, \lfloor N/2 \rfloor$ and $u_i^{(n)} = 2 - 2(n-1+u)/N = 2(N-n+1-u)/N$ for $n = \lfloor N/2 \rfloor + 1, \ldots, N$. The convergence analysis that follows is developed for standard Monte Carlo, but it can be extended to stratification and RQMC.

Let $\ell^* = \max_{\boldsymbol{\theta} \in \mathcal{S}} L(\boldsymbol{\theta})$, $\hat{\ell}_N^* = \max_{\boldsymbol{\theta} \in \mathcal{S}} \hat{L}_N(\boldsymbol{\theta})$, $S^* = \{\boldsymbol{\theta} \in \mathcal{S} : L(\boldsymbol{\theta}) = \ell^*\}$, $S_N^* = \{\boldsymbol{\theta} \in \mathcal{S} : \hat{L}_N(\boldsymbol{\theta}) = \hat{\ell}_N^*\}$. These are the optimal values of L and $\hat{L}_N$, and the sets where these optimal values are reached.

PROPOSITION 1. *For any $\mathbb{X}$, with probability 1, we have:*

(a) $\hat{L}_N$ is continuous and bounded uniformly over all possible realizations of the $u_i^{(n)}$'s, over $\mathcal{S}$;

(b) $\hat{L}_N$ converges to L uniformly over $\mathcal{S}$ when $N \to \infty$;

(c) $\hat{\ell}_N^*$ converges to ℓ^*;

(d) $\sup_{\boldsymbol{\theta} \in S_N^*} \inf_{\boldsymbol{\theta}^* \in S^*} \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \to 0$.

Proof. For (a), note that for any $\mathbb{X}$, $\varphi(\boldsymbol{\alpha}, \mathbb{X})$ is bounded over $\mathcal{S}$ and continuous in $\boldsymbol{\alpha}$. Each term of the product in (6) is continuous in α_j, λ_j, and $\bar{B}_i$, and it is also bounded over $\mathcal{S}$ by $(\alpha_j + \bar{B}_j\lambda_j)^{-\alpha_j} \leq \alpha_j^{-\alpha_j} \leq \epsilon^{-\epsilon}$. It follows that $\hat{L}_N$ is continuous and bounded over $\mathcal{S}$, uniformly over all possible realizations of the $u_i^{(n)}$'s. This proves (a). For (b), by the strong law of large numbers applied to the average in (10), we know that $\hat{L}_N(\boldsymbol{\theta}) \to L(\boldsymbol{\theta})$ w.p.1 when $N \to \infty$ for all $\boldsymbol{\theta} \in \mathcal{S}$ (point-wise convergence). To show that this convergence is uniform over $\mathcal{S}$, one can verify that the derivative of $\hat{L}_N(\boldsymbol{\theta})$ with respect to any coordinate of $\boldsymbol{\theta}$ is bounded uniformly (in absolute value) over $\mathcal{S}$ and over all possible realizations of the $u_i^{(n)}$'s. Explicit formulas for these derivatives are developed later in this Supplement, in Section A.4. Then, the family of functions $\{\hat{L}_N, N \geq 1\}$ are equicontinuous over the compact set $\mathcal{S}$, and it follows from the Arzelà-Ascoli Theorem that their sequence must converge uniformly over $\mathcal{S}$, which proves (b). Since the function L is continuous over $\mathcal{S}$, (c) and (d) then follow from Proposition 6 in Chapter 6 of Ruszczyński and Shapiro (2003).

We now know that the estimated log-likelihood function, its optimal value, and its set(s) of optimizers, converge to the exact ones, in the sense defined by the proposition. In the next sections, we develop formulas for the derivatives of $\hat{L}_N$, and then we discuss how to use these derivatives to seek an optimizer.

A.4. Derivatives of the Estimated Log-likelihood Function

We now develop formulas for the derivatives of $\hat{L}_N$ with respect to each of the parameters β, α_j, and λ_j.

By differentiating (10) with respect to α_j, we obtain

$$\begin{aligned}\frac{\partial \hat{L}_N}{\partial \alpha_j} &= \frac{\partial}{\partial \alpha_j} \log \varphi(\boldsymbol{\alpha}, \mathbb{X}) + \sum_{i=1}^{I} \frac{\partial}{\partial \alpha_j} \log \left[\frac{1}{N} \sum_{n=1}^{N} \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) \right] \\ &= \frac{\partial}{\partial \alpha_j} \log \varphi(\boldsymbol{\alpha}, \mathbb{X}) + \sum_{i=1}^{I} \frac{\sum_{n=1}^{N} \frac{\partial}{\partial \alpha_j} \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X})}{\sum_{n=1}^{N} \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X})}. \end{aligned} \tag{11}$$

The first term on the right of (11) can be computed explicitly via

$$\frac{\partial}{\partial \alpha_j} \log \varphi(\boldsymbol{\alpha}, \mathbb{X}) = I \cdot (\log \alpha_j + 1 - \Psi_0(\alpha_j)) + \sum_{i=1}^{I} \Psi_0(\alpha_j + X_{i,j}), \tag{12}$$

where $\Psi_0(\cdot)$ is the digamma function, defined as $\Psi_0(x) \stackrel{\text{def}}{=} \Gamma'(x)/\Gamma(x)$. To develop a computable expression for the derivative $\frac{\partial}{\partial \alpha_j} \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X})$ in (11), we use the fact that for any positive function f we have $\frac{\partial}{\partial x} f(x) = f(x) \frac{\partial}{\partial x} \log f(x)$, and apply this to $f(\alpha_j) = \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X})$ seen as a function of α_j only. This gives

$$\begin{aligned}\frac{\partial}{\partial \alpha_j} \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) &= \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) \frac{\partial}{\partial \alpha_j} \log \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) \\ &= \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbb{X}) \vartheta'_{i,\alpha_j}(\xi_i^{(n)}),\end{aligned}$$

where

$$\begin{aligned}\vartheta'_{i,\alpha_j}(\xi_i^{(n)}) &= \frac{\partial}{\partial\alpha_j}\log\Theta_i(\xi_i^{(n)},\boldsymbol{\alpha},\boldsymbol{\lambda},\mathbb{X})\\ &= \frac{\partial}{\partial\alpha_j}\sum_{k=1}^{p}\Big(X_{i,k}\log(\xi_i^{(n)}\lambda_k)-(X_{i,k}+\alpha_k)\log(\xi_i^{(n)}\lambda_k+\alpha_k)\Big)\\ &= -\log(\xi_i^{(n)}\lambda_j+\alpha_j)-\frac{\alpha_j+X_{i,j}}{\alpha_j+\xi_i^{(n)}\lambda_j}.\end{aligned}$$

This gives:

$$\frac{\partial\hat{L}_N}{\partial\alpha_j}=\frac{\partial}{\partial\alpha_j}\log\varphi(\boldsymbol{\alpha},\mathbb{X})+\sum_{i=1}^{I}\frac{\sum_{n=1}^{N}\vartheta'_{i,\alpha_j}(\xi_i^{(n)})\omega_i^{(n)}}{\sum_{n=1}^{N}\omega_i^{(n)}}. \tag{13}$$

The derivative of $\hat{L}_N$ with respect to λ_j is derived in a similar way. We have:

$$\frac{\partial\hat{L}_N}{\partial\lambda_j}=\sum_{i=1}^{I}\frac{\sum_{n=1}^{N}\vartheta'_{i,\lambda_j}(\xi_i^{(n)})\omega_i^{(n)}}{\sum_{n=1}^{N}\omega_i^{(n)}} \tag{14}$$

where

$$\vartheta'_{i,\lambda_j}(\xi_i^{(n)})=\frac{\partial}{\partial\lambda_j}\log\Theta_i(\xi_i^{(n)},\boldsymbol{\alpha},\boldsymbol{\lambda},\mathbb{X})=\frac{X_{i,j}}{\lambda_j}-\frac{\xi_i^{(n)}(\alpha_j+X_{i,j})}{\alpha_j+\xi_i^{(n)}\lambda_j}.$$

The derivative with respect to β can be written as

$$\frac{\partial\hat{L}_N}{\partial\beta}=\sum_{i=1}^{I}\frac{\sum_{n=1}^{N}\vartheta'_{i,\beta}(\xi_i^{(n)})\omega_i^{(n)}}{\sum_{n=1}^{N}\omega_i^{(n)}}. \tag{15}$$

where

$$\begin{aligned}\vartheta'_{i,\beta}(\xi_i^{(n)}) &= \frac{\partial}{\partial\beta}\log\Theta_i(\xi_i^{(n)},\boldsymbol{\alpha},\boldsymbol{\lambda},\mathbb{X})\\ &= \left(\frac{\partial}{\partial\xi_i^{(n)}}\log\Theta_i(\xi_i^{(n)},\boldsymbol{\alpha},\boldsymbol{\lambda},\mathbb{X})\right)\frac{\partial\xi_i^{(n)}}{\partial\beta}\\ &= \left(\sum_{j=1}^{p}\frac{X_{i,j}}{\xi_i^{(n)}}-\frac{\lambda_j(\alpha_j+X_{i,j})}{\alpha_j+\xi_i^{(n)}\lambda_j}\right)\frac{\partial G_\beta^{-1}(u_i^{(n)})}{\partial\beta},\end{aligned}$$

using the fact that $\xi_i^{(n)}=G_\beta^{-1}(u_i^{(n)})$ where $u_i^{(n)}$ is the (fixed) realization of a uniform random variable. We have no closed form formula for G_β^{-1} and no direct way of evaluating its derivative $\partial G_\beta^{-1}(u_i^{(n)})/\partial\beta$. However, it can be expressed in terms of the cdf G_β and its density g_β by using the inverse function theorem (Rudin 1976, pp. 221–223), as follows.

By differentiating the identity $G_\beta(G_\beta^{-1}(u_i^{(n)}))=u_i^{(n)}$ with respect to β, we obtain

$$\frac{\partial G_\beta}{\partial\beta}(G_\beta^{-1}(u_i^{(n)}))+\frac{\partial}{\partial x}G_\beta(x)\bigg|_{x=G_\beta^{-1}(u_i^{(n)})}\frac{\partial G_\beta^{-1}(u_i^{(n)})}{\partial\beta}=0,$$

which gives

$$\frac{\partial G_\beta^{-1}(u_i^{(n)})}{\partial \beta} = -\frac{\frac{\partial G_\beta}{\partial \beta}(G_\beta^{-1}(u_i^{(n)}))}{g_\beta(G_\beta^{-1}(u_i^{(n)}))}. \tag{16}$$

where $g_\beta(x) = \partial G_\beta(x)/\partial x = \beta^\beta x^{\beta-1} e^{-\beta x}/\Gamma(\beta)$ is the density associated with G_β. The numerator on the right side of (16) can be rewritten in terms of the lower incomplete gamma function $\gamma(\beta, x) = \int_0^x t^{\beta-1} e^{-t} dt$ (with the change of variable $t = \beta\xi_i^{(n)}$) as

$$\frac{\partial}{\partial \beta} G_\beta(\xi_i^{(n)}) = \frac{\partial}{\partial \beta} \frac{\gamma(\beta, \beta\xi_i^{(n)})}{\Gamma(\beta)} = \frac{1}{\Gamma(\beta)} \left(\frac{\partial}{\partial \beta} \gamma(\beta, \beta\xi_i^{(n)}) - \gamma(\beta, \beta\xi_i^{(n)}) \frac{\partial}{\partial \beta} \ln \Gamma(\beta) \right).$$

The derivative of the incomplete gamma function in this expression can be further developed in terms of the generalized hypergeometric function ${}_2F_2$ (Olver et al. 2010) as

$$\begin{aligned} \frac{\partial}{\partial \beta} \gamma(\beta, \beta\xi_i^{(n)}) = {} & e^{-\beta\xi_i^{(n)}} \beta^{\beta-1} {\xi_i^{(n)}}^\beta + \gamma(\beta, \beta\xi_i^{(n)}) \ln(\beta\xi_i^{(n)}) \\ & - \frac{(\beta\xi_i^{(n)})^\beta}{\beta^2} {}_2F_2(\beta, \beta; \beta+1, \beta+1; -\beta\xi_i^{(n)}). \end{aligned}$$

By plugging these expressions in (16) and then in (15), we obtain an expression for the derivative. However, this expression involves quantities that are difficult or costly to evaluate. In particular, ${}_2F_2$ is costly to evaluate for large values of β, and the asymptotic expansions we know do not work when both the argument and the parameters become large simultaneously.

For this reason, we found in our experiments that just using finite differences to approximate the derivative $\frac{\partial}{\partial \beta} G_\beta(\xi_i^{(n)})$ was faster and more robust. Another possibility could be to apply finite differences to the inverse cdf G_β^{-1} to approximate directly the derivative $\frac{\partial}{\partial \beta} G_\beta^{-1}(u_i^{(n)})$. However, the available methods we have to evaluate G_β^{-1} use root finding based on methods that evaluate G_β. In our experiments, G_β was evaluated by an approximation method for the incomplete gamma function proposed by Bhattacharjee (1970), modified to control the accuracy and implemented in SSJ (L'Ecuyer 2008).

A.5. Optimization Algorithm

Now that we know how to evaluate the smooth function $\hat{L}_N$ and its gradient at each point, we can use in principle any of the several standard first-order nonlinear optimization algorithms available to find a maximizer. Most of those algorithms are guaranteed to converge (at least) to a point where the gradient vanishes, under appropriate conditions.

In our experiments, we decided to stay away from second-order methods, which require a approximation of the Hessian (the matrix of the mixed second derivatives) at each visited point, because the gradient is already very costly to evaluate, and approximating the Hessian would require much more work. We do not believe it is worth doing. For a model with 48 periods, for example, there are

97 parameters and the Hessian has $97 \times 97 = 9409$ entries. We went the opposite way: we started by implementing a simple heuristic method which moves the coordinates one at a time, and makes the move only when the improvement is deemed good enough. We do not have a formal convergence proof for this heuristic to a global maximizer, but it cannot converge anywhere else than to a point where the gradient vanishes, and in all the examples that we tried, it always performed very well and sufficiently fast. It took between 20 minutes and two hours to estimate all parameters of a model on any given dataset.

Our heuristic method is summarized in Algorithm 1, in a generic form that maximizes a function $f(\boldsymbol{\theta}) = f(\theta_1, \ldots, \theta_s)$, in the s-dimensional real space. For our application, this f is $\hat{L}_N$. At each iteration, for each coordinate p, the algorithm computes the derivative of f at the current point with respect to coordinate p, and tests a move of that coordinate in the direction of the positive slope, by a distance equal to that derivative multiplied by a scaling factor. If the move takes the parameter outside of the box $\mathcal{S}$, then we project the move to the boundary. If the improvement from that tentative move is deemed good enough relative to the square derivative, the move is accepted (coordinate p is changed). If the relative improvement is deemed excellent, the scaling factor for p is increased (multiplied by a factor $g_1 > 1$), and if the improvement is deemed bad, the scaling factor for p is decreased (multiplied by a factor $g_0 < 1$). We repeat this for all coordinates $p = 1, \ldots, s$, in succession.

Note that when a move for coordinate p is deemed not good enough (refused), there is no change in coordinate p at this iteration. Alternatively, we could shrink the scaling factor until the move is accepted, but this may require several function evaluations to find the right factor. We tried this and it did not result in any tangible gain. We prefer to just shrink the scaling factor for this coordinate and wait for the next iteration. Our main reason for adopting a heuristic that moves one coordinate at a time is that the different coordinates generally require different scaling factors for the method to work well, because the second derivatives can be quite different along the different coordinates, and finding appropriate factors can be tedious unless we are ready to compute good estimates of the second derivatives and of the inverse Hessian, which is hardly viable given the size of problems we have to handle. Our method adjusts those factors adaptively in a very simple way.

The displayed version of the algorithm simply performs a fixed number T of iterations, and then returns an approximate optimal solution.

For the results reported in this paper, the parameter values for the algorithm were $c_0 = 0.1$, $c_1 = 0.5$, $g_0 = 1/1.21 \approx 0.826$, $c_1 = 1.1$, $T = 1000$, and we initialized $\gamma_p = 0.1$ for each p. These values worked very well for all our data sets. We also tried other combinations of parameter values from $c_0 \in [0, 0.1]$, $c_1 \in [0.3, 0.8]$, $g_0 \in (1/1.1025, 1/2.25)$, $g_1 \in (1.05, 1.5)$, $\gamma_p \in [10^{-4}, 10^{-1}]$, and they all worked almost equally well. For the initial value of $\boldsymbol{\theta}$, we used the MMEs described in Section A.1.

Algorithm 1 Coordinate-wise ascent method with adaptive scaling to maximize $f(\boldsymbol{\theta}) = f(\theta_1, \ldots, \theta_s)$

Choose an initial value for the parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_s)$;
Select initial scaling factors for the derivatives, $\gamma_1, \ldots, \gamma_s$;
Select thresholds for rescaling, $0 < c_0 < c_1$, and multipliers $0 < g_0 < 1 < g_1$;
for $t = 1$ **to** T **do**
  for $p = 1$ **to** s **do**
    Compute $f_p(\boldsymbol{\theta})$, the derivative at $\boldsymbol{\theta}$ with respect to θ_p;
    $\theta'_p \leftarrow \theta_p + \gamma_p f_p(\boldsymbol{\theta})$ {tentative update}
    $\boldsymbol{\theta}' \leftarrow (\theta_1, \ldots, \theta'_p, \ldots, \theta_s)$;
    If $\boldsymbol{\theta}' \notin \mathcal{S}$, project $\boldsymbol{\theta}'$ to $\mathcal{S}$ (take the nearest point in $\mathcal{S}$);
    $\rho_p \leftarrow [f(\boldsymbol{\theta}') - f(\boldsymbol{\theta})]/[\gamma_p f_p^2(\boldsymbol{\theta})]$; {scaled improvement from update}
    if $\rho_p > c_0$ {linear model appears sufficiently good} **then**
      $\theta_p \leftarrow \theta'_p$; {apply parameter update}
    end if;
    if $\rho_p > c_1$ {linear model looks very good} **then**
      $\gamma_p \leftarrow g_1 \gamma_p$ {expand scaling for p}
    else if $\rho_p < c_0$ {linear model looks poor} **then**
      $\gamma_p \leftarrow g_0 \gamma_p$ {shrink scaling for p}
    end if
  end for;
end for;
return approximate solution $\boldsymbol{\theta}$.

We made additional experiments to test robustness by taking 100 random initial values of $\boldsymbol{\theta}$ in a neighborhood of the MME (with up to 100% relative distance for each parameter), and also with independent realizations of the sample function $\hat{L}_N$. The parameter estimates were always very close to each other (always less than 1% difference, and much less for most parameters), and the relative difference between the largest and the smallest of the values of the likelihood at the 100 estimates was less than 10^{-6}.

A.6. Penalized Maximum Likelihood Estimation with a Smoothing Cubic Spline

We now explain how to modify the MLE computation algorithm for the version of the PG2 model where we fit a smoothing cubic spline to the gamma shape parameter α_j as a function of the successive periods j, while minimizing a criterion that accounts for both the smoothness of the spline and the quality of fit.

The smoothing cubic spline is a piecewise cubic polynomial function subject to continuity constraints between the pieces, for which the break points between the pieces (the knots) have the same abscissas as the data points (Reinsch 1967, de Boor 2001, Pollock 1993). Our spline curve S thus consists of segments S_j, each bridging the gap between adjacent knot points (j, α_j) and $(j+1, \alpha_{j+1})$, for $j = 1, \ldots, p-1$. The zero-order continuity property of the spline results in $S_j(j) = \alpha_j$ and $S_j(j+1) = \alpha_{j+1}$. There is no loss of generality in supposing that S is defined over the unity knot grid $1, 2, \ldots, p$, since the change of scale of the ordinate does not affect the overall solution. The smoothing cubic spline penalized likelihood function is defined as the weighted sum of the original log-likelihood function and the smoothness penalty term:

$$L_S = \varphi L + (1-\varphi) \int_1^p \left[\frac{\partial^2}{\partial x^2} S(x) \right]^2 dx,$$

where the weight $\varphi \in [0,1]$ determines the relative importance of the data and smoothing terms. The smoothness penalty term can be written as

$$\int_1^p \left[\frac{\partial^2}{\partial x^2} S(x) \right]^2 dx = \sum_{j=1}^{p-1} \int_j^{j+1} \left[\frac{\partial^2}{\partial x^2} S_j(x) \right]^2 dx$$

and can be further developed and computed explicitly by differentiating each cubic polynomial S_j. After some algebraic manipulations with the latter expression taking into account the smoothness constraints (see Pollock 1993 for details) and upon introducing the vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_p]$, and the following $(p-2) \times (p-2)$ matrix $\mathbf{P}$ and $(p-2) \times p$ matrix $\mathbf{F}$:

$$\mathbf{P} = \begin{bmatrix} 4 & 1 & 0 & \ldots & 0 & 0 \\ 1 & 4 & 1 & \ldots & 0 & 0 \\ 0 & 1 & 4 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & 4 \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 3 & -6 & 3 & 0 & \ldots & 0 & 0 \\ 0 & 3 & -6 & 3 & \ldots & 0 & 0 \\ 0 & 0 & 3 & -6 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & -6 & 3 \end{bmatrix},$$

we obtain the following expression for the penalized log-likelihood function:

$$L_S = \varphi L + \frac{2}{3}(1-\varphi) \boldsymbol{\alpha}^T \mathbf{F}^T \mathbf{P}^{-1} \mathbf{F} \boldsymbol{\alpha}.$$

The gradient of this function with respect to $\boldsymbol{\alpha}$ is

$$\frac{\partial}{\partial \boldsymbol{\alpha}} L_S = \varphi \frac{\partial}{\partial \boldsymbol{\alpha}} L + \frac{4}{3}(1-\varphi) \mathbf{F}^T \mathbf{P}^{-1} \mathbf{F} \boldsymbol{\alpha}.$$

The derivatives of L_S with respect to other parameters coincide with those obtained for L. The computation of the cubic spline penalized MLE can then be performed by an adaptation of the MLE methodology described in Section A.2, with the addition of the smoothness terms to both the log-likelihood function and its derivatives with respect to α_j.

Appendix B: Log-likelihood for the PG2pow Model

MLE for the PG2pow model can be done in the same way as for the PG2 model, but with a slightly different log-likelihood function. The changes include the introduction of additional gradient pre-scaling factors for the parameters p_j in the optimization algorithm, and new expressions for the derivatives with respect to β and p_j. These expressions use the following modified function Θ:

$$\Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbf{p}, \mathbb{X}) = \prod_{j=1}^{p} \frac{(\xi_i^{(n)p_j}\lambda_j/\gamma(p_j))^{X_{i,j}}}{(\alpha_j + \xi_i^{(n)p_j}\lambda_j/\gamma(p_j))^{X_{i,j}+\alpha_j}}.$$

The derivative of the estimated log-likelihood function w.r.t. the parameter β of the gamma distribution of the daily busyness factor is given by:

$$\frac{\partial \hat{L}_N}{\partial \beta} = \sum_{i=1}^{I} \frac{\sum_{n=1}^{N} \vartheta'_{i,\beta}(\xi_i^{(n)})\omega_i^{(n)}}{\sum_{n=1}^{N} \omega_i^{(n)}},$$

where by analogy $\omega_i^{(n)} = \Theta_i(\xi_i^{(n)}, \boldsymbol{\alpha}, \boldsymbol{\lambda}, \mathbf{p}, \mathbb{X})$ and

$$\vartheta'_{i,\beta}(\xi_i^{(n)}) = -\sum_{j=1}^{p} \frac{\xi_i^{(n)p_j}\lambda_j}{\gamma(p_j)} \left(X_{i,j}\left[\frac{\xi_i^{(n)p_j}\lambda_j}{\gamma(p_j)}\right]^{-1} - (X_{i,j}+\alpha_j)\left[\alpha_j + \frac{\xi_i^{(n)p_j}\lambda_j}{\gamma(p_j)}\right]^{-1}\right)\left(\frac{p_j}{\xi_i^{(n)}}\frac{\partial \xi_i^{(n)}}{\partial \beta} - \frac{\gamma'_\beta(p_j)}{\gamma(p_j)}\right),$$

where

$$\gamma'_\beta(p_j) = \frac{\beta^{-p_j}\Gamma(p_j+\beta)(-p_j/\beta + \Psi_0(p_j+\beta) - \Psi_0(\beta))}{\Gamma(\beta)}.$$

The derivative of the estimated log-likelihood function with respect to the power p_j is given by:

$$\frac{\partial \hat{L}_N}{\partial p_j} = \sum_{i=1}^{I} \frac{\sum_{n=1}^{N} \vartheta'_{i,p_j}(\xi_i^{(n)})\omega_i^{(n)}}{\sum_{n=1}^{N} \omega_i^{(n)}},$$

where

$$\vartheta'_{i,p_j}(\xi_i^{(n)}) = X_{i,j}\lambda_j \frac{\xi_i^{(n)p_j}\log[\xi_i^{(n)}\gamma(p_j)] - \gamma'_{p_j}(p_j)\xi_i^{(n)p_j}}{\gamma(p_j)^2}\left(\left[\frac{\xi_i^{(n)p_j}\lambda_j}{\gamma(p_j)}\right]^{-1} - \left[\alpha_j + \frac{\xi_i^{(n)p_j}\lambda_j}{\gamma(p_j)}\right]^{-1}\right),$$

and

$$\gamma'_{p_j}(p_j) = \frac{\beta^{-p_j}\Gamma(p_j+\beta)(-\log\beta + \Psi_0(p_j+\beta))}{\Gamma(\beta)}.$$

Appendix C: Matching Spearman Correlations in the PGnorta Model

For each pair (j,k), we need to find the correlation $\rho^{\mathrm{Z}}_{j,k} = \mathrm{Corr}(Z_j, Z_k)$ such that the Spearman correlation $r^X_{j,k} = \mathrm{Corr}(F_j(X_j), F_k(X_k))$ matches the value $\widehat{r}^X_{j,k}$ observed in the data. For this, we use Monte Carlo to estimate the (unknown) expectation in (14), and we use stochastic approximation (SA) as a root-finding method to solve this equation. See Pasupathy and Kim (2011) for a recent coverage of this SA approach.

Algorithm 2 An SA method to find appropriate correlations $\rho^{\mathrm{Z}}_{j,k}$ in the PGnorta model

Require: j,k, initial value $\rho^{\mathrm{Z}}_{j,k}(0)$;

get the MLEs of parameters α_j, for $j=1\dots p$;

for $t=1$ **to** T **do**

{generate a sample of size M for (X_j,X_k) under the PGnorta model}

for $m=1$ **to** M **do**

$[Z_j^{(m)},Z_k^{(m)}]\sim \text{Normal}\left(\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & \rho^{\mathrm{Z}}_{j,k}(t-1)\\ \rho^{\mathrm{Z}}_{j,k}(t-1) & 1\end{bmatrix}\right)$;

$U_j^{(m)}\leftarrow\Phi(Z_j^{(m)}),\quad U_k^{(m)}\leftarrow\Phi(Z_k^{(m)})$;

$B_j^{(m)}\leftarrow G^{-1}(U_j^{(m)},\,\alpha_j,\alpha_j),\quad B_k^{(m)}\leftarrow G^{-1}(U_k^{(m)},\alpha_k,\alpha_k)$;

$T_j^{(m)}\leftarrow\lambda_j B_j^{(m)},\quad T_k^{(m)}\leftarrow\lambda_k B_k^{(m)}$;

$X_j^{(m)}\sim\text{Poisson}(T_j^{(m)}),\quad X_k^{(m)}\sim\text{Poisson}(T_k^{(m)})$;

end for

$\widetilde{r}^{X}_{j,k}(t)\leftarrow\frac{1}{\widetilde{\sigma}_{\widehat{F}_j}\widetilde{\sigma}_{\widehat{F}_k}}\left[\frac{1}{M}\sum_{m=1}^{M}\widehat{F}_j(X_j^{(m)})\widehat{F}_k(X_k^{(m)})-\widetilde{\mu}_{\widehat{F}_j}\widetilde{\mu}_{\widehat{F}_k}\right]$; {sample correlation}

$\rho^{\mathrm{Z}}_{j,k}(t)\leftarrow\rho^{\mathrm{Z}}_{j,k}(t-1)+\kappa_t(\widehat{r}^{X}_{j,k}-\widetilde{r}^{X}_{j,k}(t))$; {SA iteration}

end for

Algorithm 2 summarizes our implementation. In this algorithm, κ_t is a step sequence that satisfies the conditions $\sum_{t>0}\kappa_t=\infty$ and $\sum_{t>0}\kappa_t^2<\infty$ (see Pasupathy and Kim 2011). In our experiments we used $\kappa_t=0.1t^{-\zeta}$ with $\zeta=9/16$. These constants are not hard to select in our case, because the correlation coefficient is bounded to the interval $[-1,1]$. We run the algorithm over $T=1000$ iterations. Each iteration consists of (i) generating $M=200$ samples from the NORTA model for the rate, (ii) computing the empirical Spearman correlation based on the sample counts generated in step (i), and (iii) applying one SA iteration using the difference between the Spearman correlation in the data and in the model. We initialize the algorithm by setting $\rho^{\mathrm{Z}}_{j,k}(0)\leftarrow\widehat{r}^{X}_{j,k}$.

For comparison, to estimate the copula for the model of Channouf and L'Ecuyer (2012), this algorithm would have to be modified (simplified) by taking $X_j^{(m)}=F_j^{-1}(U_j^{(m)})$ inside the "for" loop, where F_j is the cdf of $X_j^{(m)}$, which is negative binomial. However, these authors used a different approach to find the appropriate correlations for their model.

Appendix D: Bootstrap KDE Bandwidth Selection

We explain the *kernel density estimator* (KDE) method used in the bootstrapping procedures to the compute confidence intervals displayed in the plots of Section 6. Given a sample $\{x_1,\dots,x_N\}$ of size N from density $p(\cdot)$, a KDE $\widehat{p}(\cdot)$ is defined as

$$\widehat{p}(x)=\frac{1}{Nh}\sum_{i=1}^{N}k\left(\frac{x-x_i}{h}\right),$$

where $k(\cdot)$ is a suitable kernel density and h is the kernel bandwidth. It is easy to show that the mean of the KDE coincides with the empirical mean $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i$ and the variance of the KDE is related to the unbiased variance estimator $\widehat{\sigma}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \widehat{\mu})^2$ via

$$\int_{-\infty}^{\infty}(x-\widehat{\mu})^2\widehat{p}(x)dx = h^2 + \frac{N-1}{N}\widehat{\sigma}^2.$$

Thus in general the KDE variance is biased w.r.t. the sample variance. To avoid introducing this bias into the bootstrapping procedures that rely on the KDE and which are used to obtain 95% confidence intervals for quantities involving $\widehat{\sigma}^2$, we match the variance of the KDE to $\widehat{\sigma}^2$. By using the formula for the KDE variance presented above, we find that the required matching is achieved by taking the bandwidth $h = \widehat{\sigma}/\sqrt{N}$.

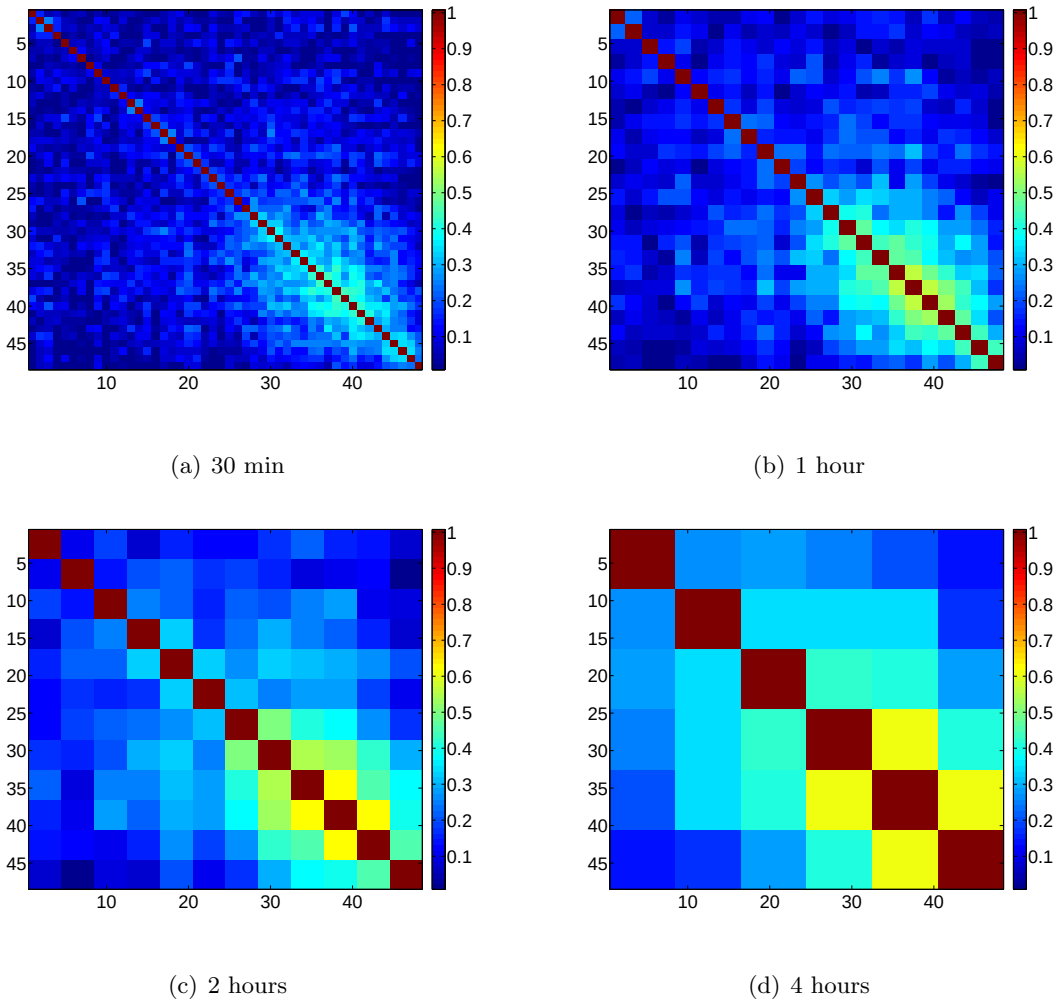
To compute confidence intervals for the DI as in Figure 3, we proceeded as follows, one period at a time. For each period p, we used the N observed counts for that period to estimate the KDE of the count. Then we resampled N independent observations from this KDE (a bootstrap sample), and computed the mean, variance, and DI from that sample. We repeated this $K = 1000$ times independently, to obtain K independent resamples of the DI, and computed the 0.025 and 0.975 quantiles of their empirical distribution. Those quantiles are the boundaries of the confidence interval.

For the confidence intervals on the correlation as in Figure 4, we used the same methodology, except that the counts X_j were replaced by pairs of counts $(Y_{1,j}, Y_{j+1,p-j})$, and the one-dimensional KDE was replaced by a two-dimensional KDE based on a two-dimensional normal kernel with mean zero and a diagonal covariance matrix with variance elements given by $1/N$ times the empirical variances of the corresponding counts. We computed $K = 1000$ bootstrap replicates of the correlation and then the 0.025 and 0.975 quantiles of the corresponding empirical distribution.

Appendix E: Additional Plots

This appendix provides additional plots for the empirical correlations observed in the data, for the three call centers. In Figures 1 to 3, the first panel shows the correlations between counts over all pairs of periods, and the other panels give the correlations between aggregated counts over blocks of 2 periods, 4 periods, and 8 periods. In the last panel, for example, each square represents a pair $(Y_{j,8}, Y_{k,8})$ where j and k are multiples of 8. The color indicates the value of the empirical correlation.

We see that the correlations are generally much smaller for first call center (emergency) than for the other two. For the emergency center, there is much more dependence between the evening periods than for the other periods of the day. In all cases, we also observe more correlation between aggregated counts than between the original counts (no aggregation). This agrees with (7).



(a) 30 min (b) 1 hour

(c) 2 hours (d) 4 hours

Figure 1 Correlations $\mathrm{Corr}(Y_{j,d}, Y_{k,d})$ for j and k multiples of d, for $d = 1, 2, 4, 8$ (30 minutes to 4 hours), for the emergency call center dataset.

References

Bhattacharjee, G. P. 1970. The incomplete gamma integral. *Applied Statistics* **19** 285–287. AS32.

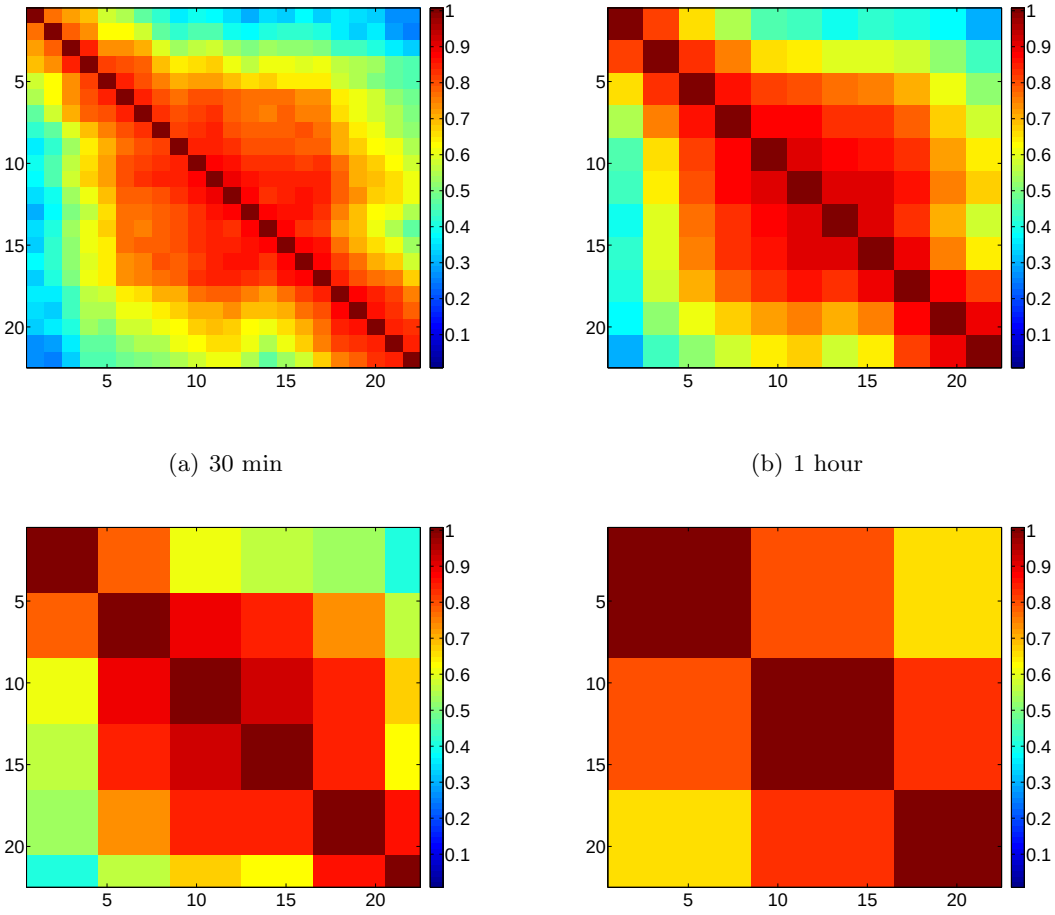
Channouf, N., P. L'Ecuyer. 2012. A normal copula model for the arrival process in a call center. *International Transactions in Operational Research* **19** 771–787.

de Boor, C. 2001. *A Practical Guide to Splines*. 2nd ed. Springer-Verlag, New York.

L'Ecuyer, P. 2008. *SSJ: A Java Library for Stochastic Simulation*. Software user's guide, available at `http://www.iro.umontreal.ca/~lecuyer`.

L'Ecuyer, P. 2009. Quasi-Monte Carlo methods with applications in finance. *Finance and Stochastics* **13**(3) 307–349.

Olver, F. W. J., D. W. Lozier, R. F. Boisvert, C. W. Clark, eds. 2010. *NIST Handbook of Mathematical Functions*. Cambridge University Press, New York, NY.



(a) 30 min (b) 1 hour

(c) 2 hours (d) 4 hours

Figure 2 Correlations $\mathrm{Corr}(Y_{j,d}, Y_{k,d})$ for j and k multiples of d, for $d = 1, 2, 4, 8$ (30 minutes to 4 hours), for the commercial call center.

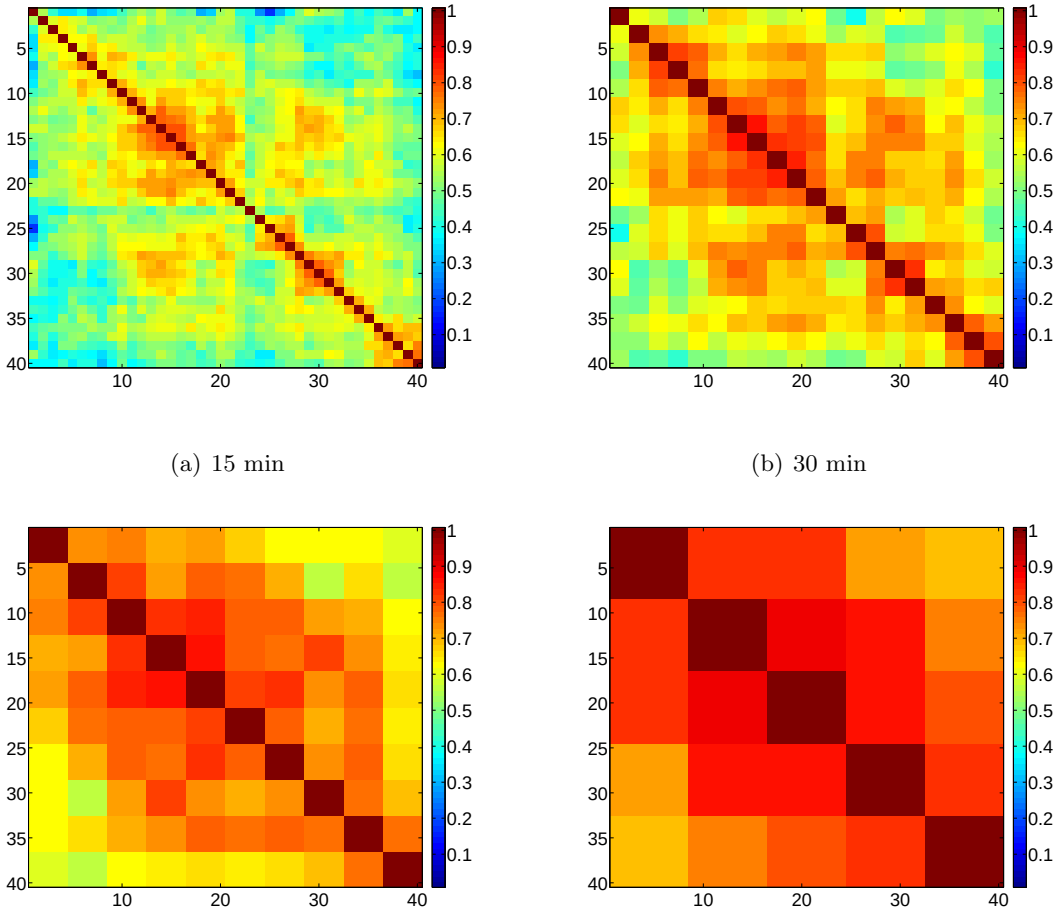
Pasupathy, R., S. Kim. 2011. The stochastic root finding problem: Overview, solutions, and open questions. *ACM Transactions on Modeling and Computer Simulation* **21**(3) Article 19.

Pollock, D. S. G. 1993. Smoothing with cubic splines. Tech. rep., University of London, Queen Mary and Westfield College, London.

Reinsch, Ch. H. 1967. Smoothing by spline functions. *Numerische Mathematik* **10** 177–183.

Rudin, W. 1976. *Principles of Mathematical Analysis*. 3rd ed. McGraw-Hill, New York.

Ruszczyński, A., A. Shapiro, eds. 2003. *Stochastic Programming*. Handbooks in Operations Research and Management Science, Elsevier, Amsterdam, The Netherlands.



(a) 15 min

(b) 30 min

(c) 1 hour

(d) 2 hours

Figure 3 Correlations $\mathrm{Corr}(Y_{j,d}, Y_{k,d})$ for j and k multiples of d, for $d = 1, 2, 4, 8$ (15 minutes to 2 hours), for the utility call center.