Multiplicative Component Analysis for replicated temporal point processes

The model

This model, proposed in Gervini (2017), sets up an additive model for the logarithm of the (unobservable) intensity functions $\lambda_1, \dots, \lambda_n$,

$$\ln \lambda_i(t) = \mu(t) + \sum_{k=1}^{p} u_{ik}\varphi_k(t)$$

where the φ_ks are orthonormal and, as a working model, we assume the component scores $u_{ik} \sim N(0, \sigma_k^2)$ independent. The component scores are treated as latent variables. This model translates into a multiplicative model for the intensities:

$$\lambda_i(t) = \lambda_0(t) \prod_{k=1}^{p} \xi_k(t)^{u_{ik}}$$

where $\lambda_0(t) = \exp \mu(t)$ is the baseline intensity and $\xi_k(t) = \exp \varphi_k(t)$ are multiplicative factors. We explain here how to estimate and interpret a multiplicative model for an Internet auction data example.

The data

The data consist of bidding times (in days) for 194 Palm Personal Digital Assistants auctioned at Ebay, and was previously analyzed by Shmueli and Jank (2005), among others. It is available in the file `Palm_7day.mat` in this package, and also at Shmueli's and Jank's personal websites.

The data is arranged as a 194 x 1 cell array `x`, where each `x{i}` is a vector with the bidding times for item *i*. So, for example, the bidding times for the 5th Palm PDA are in the vector `x{5}`.

Fitting the model

Estimation is done by the program `MCATPP`. Since the mean μ(t) and the components $\varphi_k(t)$ are modeled as spline functions, using B-spline bases with equally spaced knots, the first thing that needs to be specified are the basis parameters: the range, the number of knots and the spline order. For these data, which are weekly auctions, the range of the data is [0,7]. We use cubic splines (order 4) and 10 equally space knots, so we define

```
>> basis = struct('rng',[0 7],'or',4,'nk',10)
```

We are going to estimate, say, a two-component model, so we set `p=2`. We need to specify the smoothing parameters for the mean (`sm1`) and the components (`sm2`); two different parameters are used, in principle, because the φ_ks have unit norm but μ does not, so it may be necessary to use different smoothness parameters to attain the same degree of smoothness. The smoothness parameters can be chosen by cross-validation as follows:

```
>> [sm1,sm2,other] = cv_mcatpp(x,basis,p)
```

The parameter estimators are then computed by calling

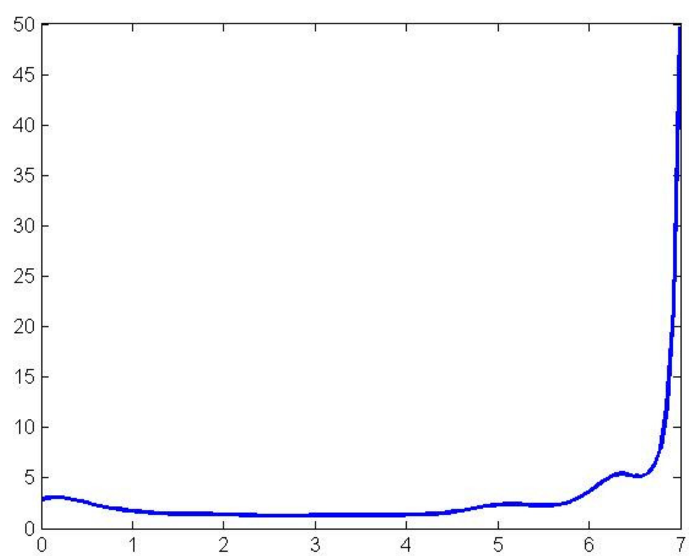
```
>> [c0,C,s2,u] = mcatpp(x,basis,p,sm1,sm2,itmax)
```

where `itmax` specifies the maximum number of iterations of the algorithm per component (we used `itmax=50`). The output vector c0 are the basis coefficients of μ(t) and the columns of C are the basis coefficients of the φ_ks. To plot these functions, as well as $\lambda_0(t)$ and the ξ_ks, we first create a grid of points and evaluate the B-spline basis functions on that grid:

```
>> t = linspace(basis.rng(1),basis.rng(2),100);
>> knots = linspace(basis.rng(1),basis.rng(2),basis.nk+2);
>> B = bspl(t,basis.or,knots,0);
```

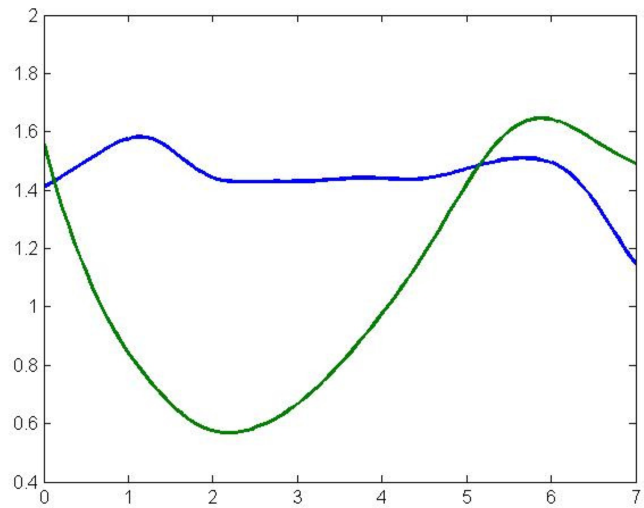
Then the estimated baseline intensity λ_0 is

```
>> plot(t,exp(B*c0),'linewidth',2)
```



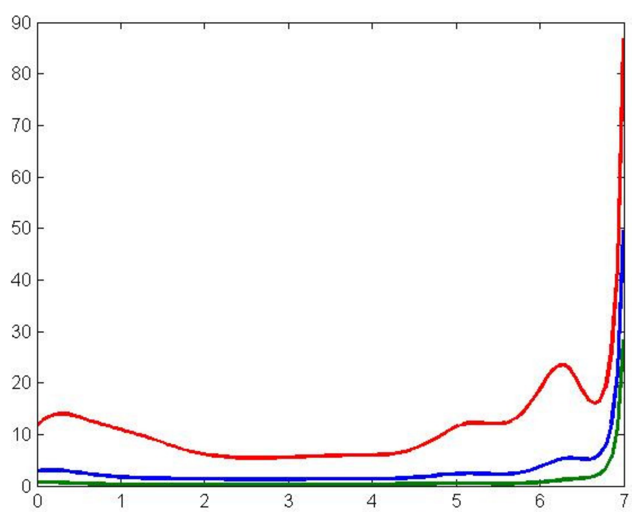
and the multiplicative components are

```
>> plot(t,exp(B*C),'linewidth',2)
```



To better interpret the components it is useful to plot the baseline $\lambda_0(t)$ versus the $\lambda(t)$s obtained by adding and subtracting a given factor (say, $2\sigma_k$) times $\varphi_k(t)$ from μ(t). For the first component, then, we obtain:

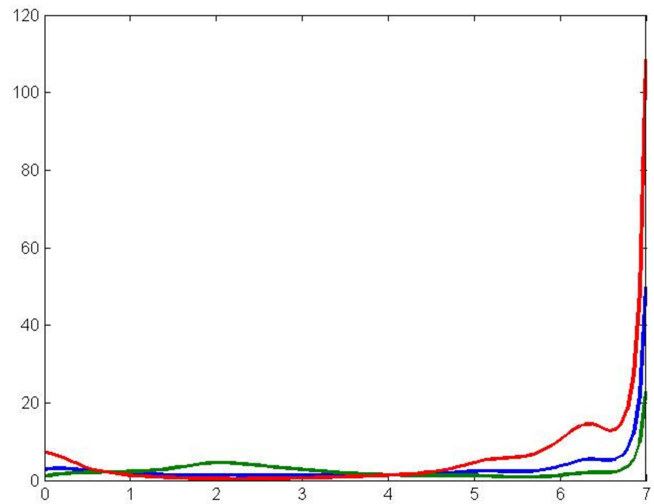
```
>> lmb0 = exp(B*c0)
>> lmbplus = exp(B*c0+2*sqrt(s2(1))*B*C(:,1))
>> lmbmin = exp(B*c0-2*sqrt(s2(1))*B*C(:,1))
>> plot(t,lmb0,t,lmbmin,t,lmbplus,'linewidth',2)
```



This shows that the first component is basically a size component, since the curves are largely parallel. So, items with large component scores u_{i1} will be items that attracted lots of bidders, while items with small u_{i1} will be those that attracted few bidders. In fact, the correlation between u_{i1} and the number of bids per item, m_i, is .88.

For the second component:

```
>> lmbplus = exp(B*c0+2*sqrt(s2(2))*B*C(:,2))
>> lmbmin = exp(B*c0-2*sqrt(s2(2))*B*C(:,2))
>> plot(t,lmb0,t,lmbmin,t,lmbplus,'linewidth',2)
```



This is a more interesting shape component, or contrast, corresponding to the phenomena known as "early biding" and "bid sniping". Items with small (negative) scores u_{i2} (green line in the plot) will tend to attract most bids (compared with the baseline) at the beginning of the auction (around the second day, specifically), whereas items with large (positive) scores u_{i2} (red line in the plot) will tend to attract most bids towards the end of the auction.

Cyclic border conditions

Sometimes the intensity functions are expected to satisfy the cyclic condition $\lambda_i(a) = \lambda_i(b)$ for time range $[a, b]$, for example, when they are daily intensities. In that case, use the program `mcatpp_cyc` for estimation and `cv_mcatpp_cyc` for cross-validation.

References

- Gervini, D. (2017). Multiplicative component models for replicated point processes. *ArXiv 1705.09693.*
- Shmueli, G., and Jank, W. (2005). Visualizing online auctions. *Journal of Computational and Graphical Statistics* **14** 299–319.