Written exercise 1

Assignment 1 – 02433 Hidden Markov Models – Anders Hørsted (s082382)

In this assignment a data set with sales figures for a soap product is analysed. The data set contains sales counts for a soap product for 242 consecutive weeks.

Question a: Choosing a model

To start the analysis of the soap sales series, a plot of the data set is created and is shown in figure 1.

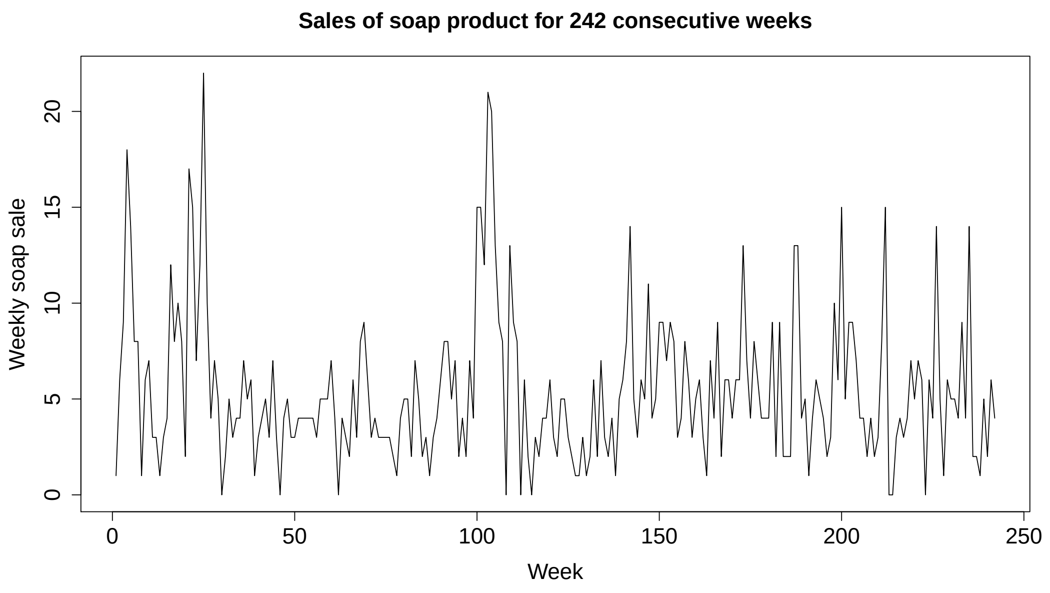

Figure 1: Weekly sales of a soap product in a supermarket for 242 consecutive weeks

From the plot it is seen that some distinct periods can be found in the data set. E.g. from week 25 to week 100 both the mean and variance seems to be smaller than for the whole data set. This could be a sign that a simple Poisson model isn't adequate to describe the data. Also the mean and variance for the whole data set is calculated, and gives

$$\bar{x} = 5.44 \qquad s^2 = 15.40$$

Since the mean equals the variance in the Poisson distribution, the data is overdispersed relative to the Poisson distribution and this further shows that a simple Poisson model will not fit the data.

One way to model data that is overdispersed relative to the simple Poisson model, is to use a independent mixture of Poisson distributions, but plotting the ACF (see figure 2) of the data, shows that consecutive data points are positive correlated. By definition, this isn't true for an independent mixture of Poisson distributions and therefore this type of model isn't considered further.

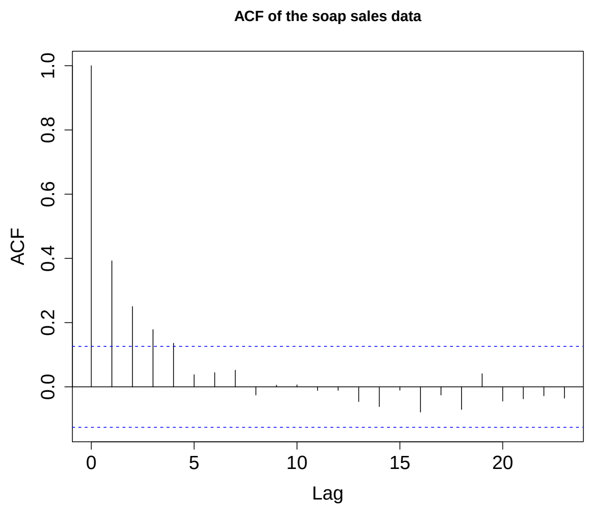

Figure 2: ACF plot of the soap data set

To handle both the overdispersed data and the correlation in the data a Poisson-HMM is instead fitted to the data.

Question b: Fitting Poisson-HMMs by direct maximization of the likelihood

In this section 2-, 3-, and 4-state Poisson-HMMs are fitted to the soap data set, by maximizing the likelihood function. A couple of details need to be handle to get a usable fit. The number of operations needed to calculate the likelihood needs to be held low. This is done by recursively calculating the forward probabilities. Also to avoid underflow problems the forward probablities need to be scaled. And to be able to use an unconstrained optimization algorithm the parameters (the Poisson means and the transition probabilities) need to be transformed to some unconstrained working parameters. All these concerns are handled by the R-function `pois.HMM.mle` in appendix A.1.4 in [3] and therefore that function is used.

Fitting a 2-state Poisson-HMM

To fit the model initial values for the parameters should be chosen. Using the advice in section 3.4.2 in [3] a single value is chosen for the off-diagonal elements in the initial t.p.m. The initial mean values are chosen symmetrically around the sample mean. As mentioned in section 3.4.1 in [3] the likelihood function will frequently have multiple local maxima so different initial values[*] need to be tested to minimize the chance of choosing a local maximum that is far from the global maximum.

For the 2-state Poisson-HMM the different initial parameters all gave the same maximum and it is therefore likely that it is indeed the global maximum. Using the initial parameters

$$\mathbf{\Gamma}_0 = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix} \qquad \boldsymbol{\lambda}_0 = \begin{pmatrix} 2 & 8 \end{pmatrix}$$

gave the following parameter estimates

$$\mathbf{\Gamma} = \begin{pmatrix} 0.912 & 0.088 \\ 0.370 & 0.630 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 4.022 & 11.371 \end{pmatrix}$$

From the estimate $\mathbf{\Gamma}$ the stationary distribution can then be calculated as

$$\boldsymbol{\delta} = \begin{pmatrix} 0.809 & 0.191 \end{pmatrix}$$

A plot of the marginal distribution of the 2-state Poisson model and a plot of the two estimated means are shown in figure 3

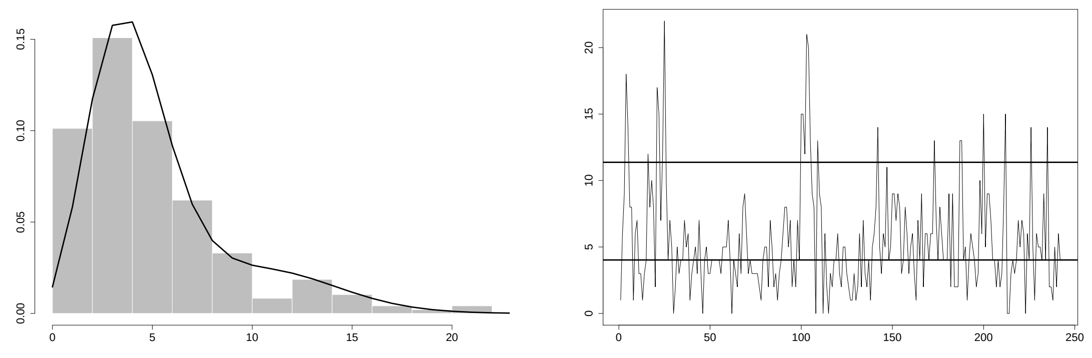

Figure 3: Left plot is a plot of the marginal distribution of the 2-state Poisson-HMM above the histogram for the soap data set. Right plot is the soap data set with the estimated means of the two state conditional distributions of the 2-state Poisson-HMM.

It is seen that the marginal distribution is matching the sample distribution rather well, and from the data plot with means included it is seen that state 1 corresponds to

[*]See appendix A.1 for the specific initial values tested

the “normal” sales level and that state 2 corresponds to the weeks with very high sale figures.

Fitting a 3-state Poisson-HMM

As with the 2-state Poisson-HMM different initial parameters[†] was used to fit a 3-state Poisson-HMM to the soap data set. Two different local maxima was found depending on the initial values of the $\boldsymbol{\lambda}$ parameter. For the lowest of the two maxima one of the state dependent means was close to 0 which gives a model that is almost a 2-state HMM. For the other maximum all state dependent means were different from zero, and this maximum was found by eg. using the intial parameters

$$\boldsymbol{\Gamma}_0 = \begin{pmatrix} 0.8 & 0.1 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{pmatrix} \qquad \boldsymbol{\lambda}_0 = \begin{pmatrix} 2 & 5 & 8 \end{pmatrix}$$

Which gave the estimates

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.864 & 0.117 & 0.019 \\ 0.445 & 0.538 & 0.017 \\ 0.000 & 0.298 & 0.702 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 3.736 & 8.443 & 14.927 \end{pmatrix}$$

and the stationary distribution

$$\boldsymbol{\delta} = \begin{pmatrix} 0.722 & 0.220 & 0.058 \end{pmatrix}$$

It is seen that the process is only in state 3 in little more than 5% of the time. With a mean of $\lambda_3 = 14.927$ this state handles only the few weeks with exceptionally high sale figures. This is also seen in the data plot with state dependent means shown in figure 4.

[†]See appendix A.2 for the initial values that was tested

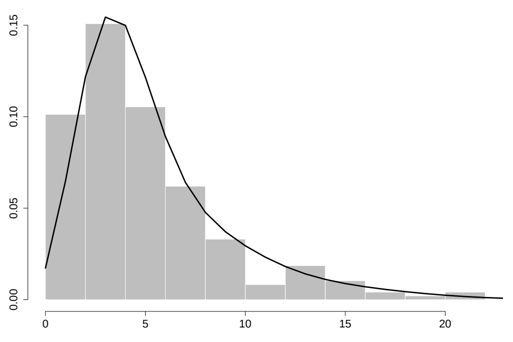
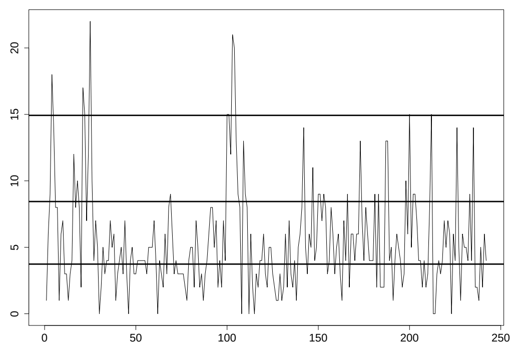

Figure 4: Left plot is a plot of the marginal distribution of the 3-state Poisson-HMM above the histogram for the soap data set. Right plot is the soap data set with the estimated means of the two state conditional distributions of the 3-state Poisson-HMM.

Fitting a 4-state Poisson-HMM

As for the 2- and 3-state models different initial values are used when fitting the 4-state model, but unlike the 2- and 3-state models, the maxima obtained for the 4-state model is quite sensitive to the initial parameters. This is seen in table 1 where 9 different initial values are used to fit a 4-state model.

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.008 & 8.226 & 3.886 & 14.359\end{pmatrix}$	604.26
0.05	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.000 & 3.873 & 8.197 & 14.690\end{pmatrix}$	604.19
0.10	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.000 & 8.197 & 3.873 & 14.690\end{pmatrix}$	604.19
0.01	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}3.063 & 3.995 & 6.069 & 12.034\end{pmatrix}$	612.28
0.05	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}4.243 & 3.490 & 8.952 & 14.788\end{pmatrix}$	608.80
0.10	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}3.911 & 0.415 & 8.192 & 14.627\end{pmatrix}$	604.49
0.01	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.019 & 7.594 & 3.789 & 13.188\end{pmatrix}$	605.68
0.05	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.000 & 3.873 & 8.197 & 14.690\end{pmatrix}$	604.19
0.10	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.000 & 3.873 & 8.197 & 14.690\end{pmatrix}$	604.19

Table 1: Maximum likelihood for 9 different starting values for the 4-state Poisson-HMM

As with the 3-state model some of the found maxima have $\lambda_1 < 10^{-10}$ which effectively makes it a 3-state model. Ignoring the maxima with λ_1 very small the model is found with the initial parameters

$$\mathbf{\Gamma}_0 = \begin{pmatrix} 0.97 & 0.01 & 0.01 & 0.01 \\ 0.01 & 0.97 & 0.01 & 0.01 \\ 0.01 & 0.01 & 0.97 & 0.01 \\ 0.01 & 0.01 & 0.01 & 0.97 \end{pmatrix} \qquad \boldsymbol{\lambda}_0 = \begin{pmatrix} 1 & 4 & 6 & 9 \end{pmatrix}$$

The estimates found are

$$\mathbf{\Gamma} = \begin{pmatrix} 0.173 & 0.000 & 0.640 & 0.187 \\ 0.082 & 0.580 & 0.338 & 0.000 \\ 0.000 & 0.092 & 0.882 & 0.025 \\ 0.002 & 0.317 & 0.000 & 0.682 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 0.008 & 8.226 & 3.886 & 14.359 \end{pmatrix}$$

and it is seen 4 of 16 elements in $\mathbf{\Gamma}$ is smaller than $5 \cdot 10^{-4}$ which isn't surprising as explained on page 52 in [3]. Also $\lambda_1 = 0.008$ is mathematically different from 0, but a Poisson distribution with $\lambda = 0.008$ will only with very small probability be different from 0[‡]. Ignoring state 1 it is seen that the means of state 2,3,4 matches the means of state 1,2,3 in the 3-state model. From $\mathbf{\Gamma}$ the stationary distribution is found as

$$\boldsymbol{\delta} = \begin{pmatrix} 0.021 & 0.207 & 0.704 & 0.068 \end{pmatrix}$$

Which further confirms that state 1 is handling special cases. Plots of the marginal distribution and the state dependent means are shown in figure 5.

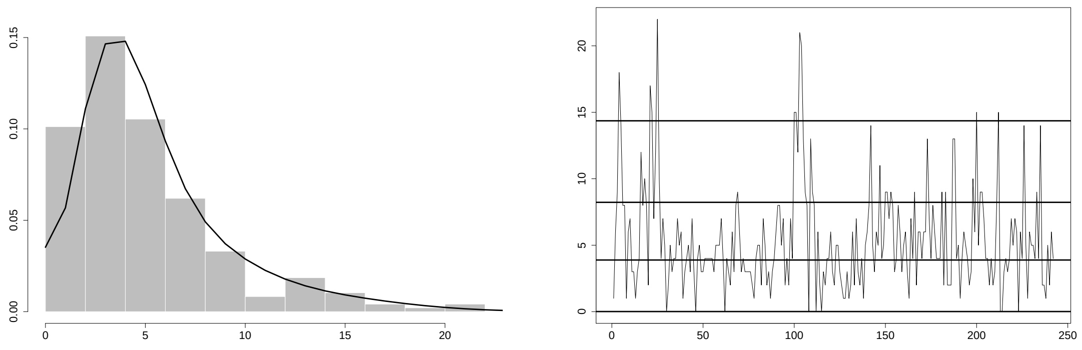

Figure 5: Left plot is a plot of the marginal distribution of the 4-state Poisson-HMM above the histogram for the soap data set. Right plot is the soap data set with the estimated means of the two state conditional distributions of the 4-state Poisson-HMM.

The marginal distribution of the 4-state model is very much like the marginal distribution of the 3-state model and the question is whether the extra parameters in the 4-state model are needed

[‡]A quick idea of how small the probability is can be seen from Chebyshevs inequality (see eg. [2]) which gives that $\mathrm{P}(|X - E[X]| \geq 1) \leq 0.008$ when $X \sim Pois(0.008)$

Model selection

To choose between the 2-, 3- and 4-state model the AIC and the BIC of the models can be compared. Furthermore the mean, variance and correlation of the models can be compared with the sample mean, variance and correlation. To calculate the mean, variance and correlations for the models, the function `pois.HMM.moments` from appendix A.7, is used.

	μ	σ^2	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	ρ_7
Sample	5.442	15.401	0.392	0.250	0.178	0.136	0.038	0.044	0.052
2-states	5.429	13.784	0.329	0.178	0.097	0.052	0.028	0.015	0.008
3-states	5.421	14.721	0.407	0.268	0.178	0.120	0.081	0.055	0.037
4-states	5.418	14.779	0.380	0.241	0.157	0.103	0.067	0.044	0.029

Table 2: Mean, variance and correlation coefficients for the data and the three models. Note that $\rho_k = Corr(X_t, X_{t+k})$

	AIC	BIC
2-states	1245.337	1259.292
3-states	1239.043	1270.444
4-states	1240.528	1296.351

Table 3: AIC and BIC scores for the three models

From the results in table 2 it is seen that all three model means are close to the sample mean. The variance of the 2-state model is lower than the sample variance and the correlation coefficients of the 2-state model are all lower than the sample correlation coefficients. On the other hand both the 3- and 4-state models seems to match both the variance and correlation coefficients of the sample pretty well. To choose between the 3- and 4-state model the AIC and BIC score are therefore used.

From table 3 it is seen that the 3-state model scores better than the 4-state model in both AIC and BIC score. This indicates that a 4-state model might overfit the data. It is worth noting that the 2-state model has the lowest BIC of the three models. This is due to the fact that the BIC penalizes models with many parameters more[§] than the AIC score does. The low variance and correlation coefficients of the 2-state model, compared with the sample, rules out the 2-state model though and the final choice of model is the 3-state model.

Question c: Fitting Poisson-HMMs using the EM algorithm

In this section 2-, 3- and 4-state Poisson-HMM models are fitted to the soap data set using the EM algorithm. The markov chain is not assumed to be stationary. As mentioned in section 4.2.5 in [3] this makes the implementation of the EM algorithm easier.

[§]Whenever the number of observations is larger than $e^2 \approx 7$

The distribution $\boldsymbol{\delta}_0$ of the first state then needs to be determined; either by including the distribution in the parameters to estimate or by maximizing the likelihood conditioned on the Markov chain starting in a particular state ($\boldsymbol{\delta}_0$ being a unitvector). As mentioned in section 4.2.4 the easiest approach is to maximize the likelihood conditioned on the Markov chain starting in a particular state, but since the implementation in appendix A.2.3 in [3] estimates $\boldsymbol{\delta}_0$ that is the approach taken in this assignment. As in the examples in section 4.3 in [3] the starting values for $\boldsymbol{\delta}_0$ is set as the uniform distribution.

Fitting a 2-state Poisson-HMM

Using the same 9 initial parameter values as in the direct maximization of the likelihood the 2-state Poisson-HMM parameters was estimated by the EM algorithm. All 9 starting points gave the same maximum, so it is likely to be a global maximum. The estimated parameters was found as

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.911 & 0.089 \\ 0.367 & 0.633 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 4.019 & 11.354 \end{pmatrix}$$

$$\boldsymbol{\delta}_0 = \begin{pmatrix} 1.000 & 0.000 \end{pmatrix}$$

It is seen that the distribution of the initial state is found as a unit vector which following section 4.2.4 in [3] is to be expected at a maximum. It is also worth noticing that the found parameters for $\boldsymbol{\Gamma}$ and $\boldsymbol{\lambda}$ are almost the same as those found by direct maximization of the likelihood function. Consulting table 7 in appendix A.3 shows that the maximum likehood value is slightly higher for the EM algorithm than for the direct maximization. As mentioned in section 4.3.1 in [3] this is also expected since the initial distribution was constrained to be the stationary distribution when fitting by direct maximization.

Fitting a 3-state Poisson-HMM using the EM algorithm

Using the same initial parameters as for the direct maximization method and using a uniform distribution as the starting value for the initial distribution $\boldsymbol{\delta}_0$, a 3-state Poisson-HMM was fitted to the soap data by the EM algorithm. All results are shown in table 8 in appendix A.4 and it is seen that two distinct maxima was found. The best maximum found had the parameters

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.861 & 0.120 & 0.019 \\ 0.440 & 0.542 & 0.017 \\ 0.000 & 0.295 & 0.705 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 3.725 & 8.397 & 14.916 \end{pmatrix}$$

$$\boldsymbol{\delta}_0 = \begin{pmatrix} 1.000 & 0.000 & 0.000 \end{pmatrix}$$

Again the estimated initial distribution is seen to be a unit vector as expected. Also the found parameters are almost identical to the parameters found by direct maximization and the likelihood value is a bit higher for the maximum found be the EM algorithm.

Fitting a 4-state Poisson-HMM using the EM algorithm

Using the same starting values as for the direct maximization and using a uniform distribution for the starting value of the initial state distribution a 4-state Poisson-HMM was fitted. All results are shown in table 9 in appendix A.5 and it is seen that only 3 different maxima was found. Using the direct maximization method gave 6 different maxima for the 4-state model so for this particular data set the EM algorithm seems to be less sensitive to the starting values. The best maximum was found with the parameters given by

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.100 & 0.616 & 0.000 & 0.285 \\ 0.000 & 0.895 & 0.090 & 0.015 \\ 0.131 & 0.261 & 0.608 & 0.000 \\ 0.000 & 0.000 & 0.285 & 0.715 \end{pmatrix} \qquad \boldsymbol{\lambda} = \begin{pmatrix} 0.619 & 3.932 & 8.252 & 14.101 \end{pmatrix}$$

$$\boldsymbol{\delta}_0 = \begin{pmatrix} 1.000 & 0.000 & 0.000 & 0.000 \end{pmatrix}$$

As expected the estimated initial state distribution is a unit vector. The log likelihood value at the maximum is $\ell_{\text{em}} = -602.77$ which is higher than the maximum value $\ell_{\text{dir}} = -604.19$ found using the direct maximization method.

Comparing the direct maximization method with the EM algorithm

In this section a few comments about the pros and cons of the direct maximization method as well as the EM algorithm is made.

Since implementations of both methods are given in [3] the work involved in fitting Poisson-HMMs using the 2 methods was more or less the same. Also since the data set is rather small, no significant differences in the performances of the two algorithms was noticed. For some initial parameter values, `NA` values were produced by the `nlm` method when using the direct maximization method. Since the EM algorithm for a Poisson-HMM do not rely on numerical optimization this wasn't a problem for the EM algorithm. Another point already mentioned in a previous section is the fact that the EM algorithm did find maxima with slightly higher likelihood values, since it do not assume stationary distribution of the Markov Chain. Based on the concrete experience of fitting the soap data to 2-, 3- and 4-state Poisson-HMMs the preferred algorithm would probably be the EM algorithm.

Since the soap data set is the only data set I have ever fitted to a HMM, the previous choice of algorithm isn't based on any more general experiences with HMMs. Reading the discussions in section 4.4 in [3] and section 10.1.4 in [1] it seems that the direct maximization method is converging faster. And with the high availability of robust numerical optimization algorithms, much of the work in implementing the direct maximization method is already taken care of. To sum up the discussions. If the M-step in the EM algorithm do not have a analytical solution, the direct maximization method is preferred, but if the M-step do have an analytical solution the choice of algorithm is still debatable.

Question d: Local and global decoding and state prediction

In this section a local decoding is applied to the 3-state Poisson-HMM fitted by the EM algorithm. For each week t the maximum of $\Pr(C_t = i \mid \boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)})$ across the states $i = 1, 2, 3$ is found using equation 5.6 in [3]. An implementation that tries to avoid underflow is given in A.2.7 in [3] and that is what is used. The result is given in figure 6

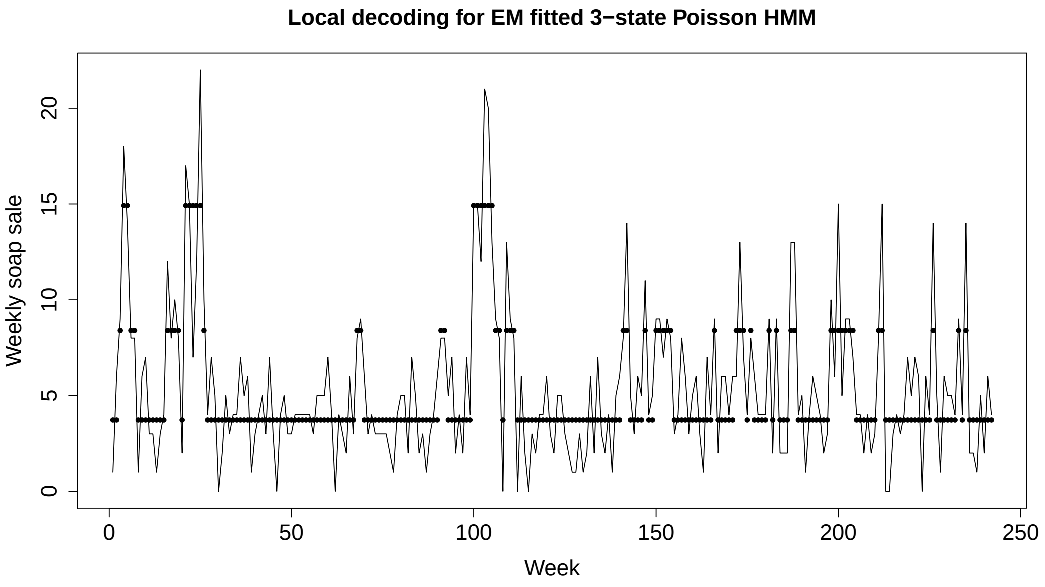

Figure 6: The local decoded states of the fitted 3-state Poisson-HMM

When the states in the HMM represents something meaningful it may be more interesting to find the sequence of states that maximizes $\Pr(\boldsymbol{C}^{(T)} = \boldsymbol{c}^{(T)} \mid \boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)})$. Whether this is the case for the Poisson-HMM model of the soap sales is not entirely clear, but using the Viterbi algorithm (as implemented in A.2.4 in [3]) a global decoding of the 3-state fitted Poisson-HMM is applied giving the result in figure 7.

The difference between the local and global decoding is minor. Running

```
sum(1*(local.decoding!=global.decoding))
```

shows that only 7 weeks differs between the local and global decoding.

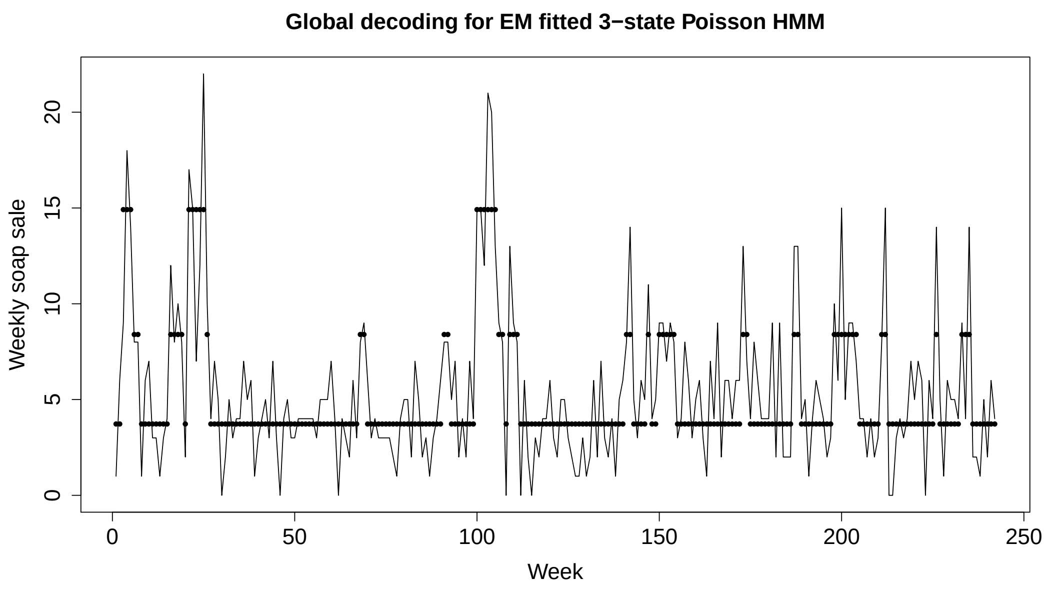

Figure 7: The global decoded states of the fitted 3-state Poisson-HMM

State prediction

As a final task the h-step-ahead predictions for $h = 1, 2, 3, 4, 5$ are calculated for the 3-state Poisson-HMM fitted by the EM algorithm. The state predictions are calculated using the R function in A.2.7 in [3] and the results are shown in table 4

h	State 1	State 2	State 3
1	0.839	0.142	0.019
2	0.785	0.183	0.032
3	0.756	0.203	0.041
4	0.740	0.213	0.047
5	0.731	0.218	0.051

Table 4: h-step-ahead state predictions for $h = 1, 2, 3, 4, 5$

Comparing the results with the stationary distribution of the 3-state model

$$\boldsymbol{\delta} = \begin{pmatrix} 0.714 & 0.226 & 0.060 \end{pmatrix}$$

it is seen that the h-step-ahead state distribution is converging towards the stationary distribution. This is as expected as mentioned on the bottom of page 86 in [3].

A Appendices

All R code created for this assignment is included here. All source code incl. latex code for this report can be found at `https://github.com/alphabits/dtu-spring-2012/tree/master/02433/assignment-1`

A.1 Fitting 2-state Poisson-HMM by direct maximization of MLE

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.05	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.10	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.01	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.05	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.10	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.01	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.05	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67
0.10	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.022 & 11.371\end{pmatrix}$	618.67

Table 5: Maximum likelihood for 9 different starting values for the 2-state Poisson-HMM

A.2 Fitting 3-state Poisson-HMM by direct maximization of MLE

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.000 & 4.134 & 11.442\end{pmatrix}$	612.14
0.05	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.000 & 4.134 & 11.442\end{pmatrix}$	612.14
0.10	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.000 & 4.134 & 11.442\end{pmatrix}$	612.14
0.01	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}3.736 & 8.443 & 14.927\end{pmatrix}$	610.52
0.05	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}3.736 & 8.443 & 14.927\end{pmatrix}$	610.52
0.10	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}3.736 & 8.443 & 14.927\end{pmatrix}$	610.52
0.01	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}8.443 & 3.736 & 14.927\end{pmatrix}$	610.52
0.05	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}3.736 & 8.443 & 14.927\end{pmatrix}$	610.52
0.10	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}3.736 & 8.443 & 14.927\end{pmatrix}$	610.52

Table 6: Maximum likelihood for 9 different starting values for the 3-state Poisson-HMM

A.3 Fitting 2-state Poisson-HMM with the EM algorithm

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.05	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.10	$\begin{pmatrix}2 & 8\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.01	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.05	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.10	$\begin{pmatrix}3 & 7\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.01	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.05	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45
0.10	$\begin{pmatrix}4 & 6\end{pmatrix}$	$\begin{pmatrix}4.019 & 11.354\end{pmatrix}$	618.45

Table 7: Maximum likelihood for 9 different starting values for the 2-state Poisson-HMM

A.4 Fitting 3-state Poisson-HMM with the EM algorithm

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.371 & 4.142 & 11.304\end{pmatrix}$	611.29
0.05	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.371 & 4.142 & 11.304\end{pmatrix}$	611.29
0.10	$\begin{pmatrix}1 & 5 & 9\end{pmatrix}$	$\begin{pmatrix}0.371 & 4.142 & 11.304\end{pmatrix}$	611.29
0.01	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}3.725 & 8.397 & 14.916\end{pmatrix}$	610.20
0.05	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}0.371 & 4.142 & 11.304\end{pmatrix}$	611.29
0.10	$\begin{pmatrix}2 & 5 & 8\end{pmatrix}$	$\begin{pmatrix}0.371 & 4.142 & 11.304\end{pmatrix}$	611.29
0.01	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}3.725 & 8.397 & 14.916\end{pmatrix}$	610.20
0.05	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}3.725 & 8.397 & 14.916\end{pmatrix}$	610.20
0.10	$\begin{pmatrix}3 & 5 & 7\end{pmatrix}$	$\begin{pmatrix}3.725 & 8.397 & 14.916\end{pmatrix}$	610.20

Table 8: Maximum likelihood for 9 different starting values for the 3-state Poisson-HMM

A.5 Fitting 4-state Poisson-HMM with the EM algorithm

$\gamma_{ij,i\neq j}$	$\boldsymbol{\lambda}_0$	$\boldsymbol{\lambda}$	$-\ell$
0.01	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.272 & 3.805 & 7.875 & 12.928\end{pmatrix}$	604.70
0.05	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.10	$\begin{pmatrix}1 & 4 & 6 & 9\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.01	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}0.522 & 3.909 & 5.894 & 11.817\end{pmatrix}$	606.28
0.05	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.10	$\begin{pmatrix}2 & 4 & 6 & 8\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.01	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.05	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77
0.10	$\begin{pmatrix}1 & 4 & 8 & 12\end{pmatrix}$	$\begin{pmatrix}0.619 & 3.932 & 8.252 & 14.101\end{pmatrix}$	602.77

Table 9: Maximum likelihood for 9 different starting values for the 4-state Poisson-HMM

A.6 R code for loading data

```
data = read.table("data/soap.txt")
x = data[,1]
mean.x = mean(x)
var.x = var(x)
```

A.7 Main function file

```r
source('src/utils.R')
source('src/export-helpers.R')
source('src/plot-helpers.R')


statdist = function (gamma) {
    m = dim(gamma)[1]
    one.row = rep(1, m)
    U = matrix(rep(one.row, m), nrow=m)
    Id = diag(one.row)
    inv = solve(Id - gamma + U)
    return(one.row %*% inv)
}

p.HMM = function (x, lambda, gamma, delta=FALSE) {
    if (delta == FALSE) {
        delta = statdist(gamma)
    }

    return(sum(dpois(x, lambda=lambda)*delta))
}

# From exercise 9 in week 2
pois.HMM.moments = function (m, lambda, gamma, lag.max=10) {
    delta = statdist(gamma)
    lambda = matrix(lambda)
    Lambda = diag(c(lambda))
    expectation = delta %*% lambda
    expectation.sq = expectation^2
    M = delta %*% Lambda %*% lambda
    variance = expectation + M - expectation.sq
    correlations = matrix(c(1, rep(0, lag.max)), nrow=1)
    tmp.gamma = gamma
    for (i in 2:(lag.max+1)) {
        cov = delta %*% Lambda %*% tmp.gamma %*% lambda - expectation.sq
        correlations[i] = cov/variance
        tmp.gamma = tmp.gamma %*% gamma
    }

    return(list(
        expectation=expectation,
        variance=variance,
        correlations=correlations
    ))
}

get.initial.gamma = function (m, off.diag=0.1) {
    on.diag = 1 - (m-1)*off.diag
    dat = c()
    for (column in 1:m) {
        if (column > 1) {
            dat = c(dat, rep(off.diag, column-1))
        }
        dat = c(dat, on.diag)
        if (column < m) {
            dat = c(dat, rep(off.diag, m-column))
        }
    }
    return(matrix(dat, nrow=m))
}

calculate.fit.result = function (fit.func, lambdas, off.diags, m, ...) {
    fit.res = list()
    for (lambda in lambdas) {
        for (off.diag in off.diags) {
            gamma.init = get.initial.gamma(m, off.diag)
            fit = fit.func(x, m, lambda, gamma.init, ...)
            fit$lambda.init = lambda
            fit$gamma.init = gamma.init
            fit$moments = pois.HMM.moments(m, fit$lambda, fit$gamma)
            fit.res[[(length(fit.res)+1)]] = fit
        }
    }
    return(fit.res)
}

calculate.fit.result.ml = function (lambdas, off.diags, m) {
    return(calculate.fit.result(pois.HMM.mle, lambdas, off.diags, m))
}

calculate.fit.result.em = function (lambdas, off.diags, m, ...) {
    return(calculate.fit.result(pois.HMM.EM, lambdas, off.diags, m, ...))
}
```

A.8 Utility functions

```
plot.and.save = function(filename, width, height, cex, plotfunction, ...) {
    save.the.plot = exists('SAVEPLOTS') && SAVEPLOTS
    if (save.the.plot) {pdf(sprintf('plots/%s', filename), width, height)}
    plotfunction(..., cex.lab=cex, cex.main=cex, cex.axis=cex)
    if (save.the.plot) {dev.off()}
}

float.str = function (float) {
    return(sprintf('%.03f', float))
}
```

A.9 Functions to export results to latex

```
save.fit.results = function (fit, label) {
    for (key in c('delta', 'gamma', 'lambda')) {
        save.latex.matrix(sprintf('%s-%s.tex', label, key), fit[[key]])
    }
}

repeated.fit.results.table = function (fits) {
    num.fits = length(fits)
    output.rows = rep("", num.fits)
    for (i in 1:num.fits) {
        fit = fits[[i]]
        output.rows[i] = sprintf('$%.02f$ & $%s$ & $%s$ & $%.02f$',
                                 fit$gamma.init[2,1],
                                 latex.matrix(fit$lambda.init, '%i'),
                                 latex.matrix(fit$lambda),
                                 fit$mllk)
    }
    return(paste(output.rows, collapse=" \\\\\n"))
}

save.repeated.fit.results = function (fits, label) {
    sink(sprintf('results/%s-fit-table.tex', label))
    cat(repeated.fit.results.table(fits))
    sink()
}

save.comparison.table = function (filename, sample, fits) {
    path = sprintf('results/%s', filename)
    sink(path)
    cat(get.comparison.table(sample, fits))
    sink()
}

get.comparison.table = function (sample, fits) {
    sample.corr = acf(sample, plot=F)[[1]]
    sample.mean = mean(sample)
    sample.var = var(sample)
    output.rows = c(get.comparison.table.row('Sample', sample.corr, sample.mean,
                                             sample.var))
    for (fit.num in 1:length(fits)) {
        label = paste(fit.num+1, '-states', sep="")
        fit = fits[[fit.num]]
        output.rows = c(output.rows,
                        get.comparison.table.row(label,
                                                 fit$moments$correlations,
                                                 fit$moments$expectation,
                                                 fit$moments$variance))
    }
    return(paste(output.rows, collapse="\n"))
}

get.comparison.table.row = function (label, corrs, mean, var) {
    return(paste(
        label, '&',
        float.str(mean), '&',
        float.str(var), '&',
        paste(float.str(corrs[2:8]), collapse=" & "), "\\\\"
    ))
}
```

```
get.information.content.table = function (fits) {
    rows = c()
    for (fit.num in 1:length(fits)) {
        fit = fits[[fit.num]]
        label = paste(fit.num+1, '-states', sep='')
        rows = c(rows, paste(label, '&',
                             float.str(fit$AIC), '&',
                             float.str(fit$BIC), "\\\\"))
    }
    return(paste(rows, collapse="\n"))
}

save.information.content.table = function (filename, fits) {
    path = sprintf('results/%s', filename)
    sink(path)
    cat(get.information.content.table(fits))
    sink()
}

plot.fit.results = function (x, fit, label) {
    lambda = fit$lambda
    gamma = fit$gamma
    plot.and.save(sprintf('%s-%s.pdf', label, 'histogram'), 12, 9, 1.5,
                  plot.HMM.with.hist, x, lambda, gamma, main="", xlab="",
                  ylab="")
    plot.and.save(sprintf('%s-%s.pdf', label, 'means-on-data'), 12, 9, 1.5,
                  plot.data.with.means, x, lambda, main="", xlab="",
                  ylab="")
}

get.ys.for.state.plot = function (states, lambda) {
    ys = rep(0, length(states))
    for (state.num in 1:length(lambda)) {
        active.points = states==state.num
        ys = ys + lambda[state.num]*active.points
    }
    return(ys)
}

latex.matrix = function (mat, digit.format='%.03f') {
    if (!class(mat) == "matrix") {
        mat = matrix(mat, ncol=length(mat))
    }
    return(paste("\\begin{pmatrix}",
                 latex.table.content(mat, digit.format),
                 "\\end{pmatrix}",
                 sep="\n"))
}

latex.table.content = function (mat, digit.format='%.03f') {
    row.output = c()
    for (i in 1:dim(mat)[1]) {
        row.output[i] = paste(sprintf(digit.format, mat[i,]), collapse=" & ")
    }
    return(paste(row.output, collapse="\\\\"))
}

save.latex.matrix = function (filename, mat, digit.format='%.03f') {
    sink(sprintf('results/%s', filename))
    cat(latex.matrix(mat, digit.format))
    sink()
}

get.state.prediction.table = function (preds) {
    output = c()
    for (i in 1:dim(preds)[1]) {
        output[i] = paste(i, " & ",
                          paste(float.str(preds[i,]), collapse=" & "))
    }
    return(paste(output, collapse="\\\\"))
}
```

A.10 Functions for plotting

```
plot.predicted.states = function (x, states, lambda, ...) {
    plot.ys = get.ys.for.state.plot(states, lambda)
    plot(x, type="l", ...)
    points(plot.ys, pch=20)
}

plot.HMM = function (lambda, gamma, max.x=25, plotfns=plot, ...) {
    delta = statdist(gamma)
    x = 0:max.x
    y = sapply(x, p.HMM, lambda=lambda, gamma=gamma, delta=delta)
    plotfns(x, y, ...)
}

plot.HMM.with.hist = function (x, lambda, gamma, ...) {
    hist(x, freq=F, ylim=c(0, 0.16), col="gray", border="white", ...)
    plot.HMM(lambda, gamma, plotfns=lines, lwd=3)
}

plot.data.with.means = function (x, means, ...) {
    plot(x, type="l", ...)
    for (m in means) {
        lines(c(-1000, 1000), rep(m, 2), lwd=3)
    }
}
```

A.11 Plotting the data

```
# All paths are relative to assignment root dir
source('src/loaddata.R')
source('src/functions.R')

SAVEPLOTS = TRUE

plot.and.save("sales-series.pdf", 12, 7, 1.5, plot,
    x, type="l", xlab="Week", ylab="Weekly soap sale",
    main="Sales of soap product for 242 consecutive weeks")

plot.and.save("acf-data.pdf", 8, 7, 1.5, acf, x,
              main="ACF of the soap sales data")
```

A.12 Fitting models by direct maximization

```
# All paths are relative to assignment root
source('src/loaddata.R')
source('src/functions.R')
source('src/A1.R')
source('src/A2.R')

SAVEPLOTS = TRUE

off.diag.to.test = c(0.01, 0.05, 0.1)


lambdas.2 = list(c(2,8), c(3,7), c(4,6))
fit.2.res = calculate.fit.result.ml(lambdas.2, off.diag.to.test, 2)
# One fit chosen for the report
fit.2 = fit.2.res[[3]]

lambdas.3 = list(c(1,5,9), c(2,5,8), c(3,5,7))
fit.3.res = calculate.fit.result.ml(lambdas.3, off.diag.to.test, 3)
# One fit chosen for the report
fit.3 = fit.3.res[[6]]

lambdas.4 = list(c(1, 4, 6, 9), c(2, 4, 6, 8), c(1, 4, 8, 12))
fit.4.res = calculate.fit.result.ml(lambdas.4, off.diag.to.test, 4)
# One fit chosen for the report
fit.4 = fit.4.res[[1]]
```

A.13 Fitting models by EM algorithm

```r
# All paths are relative to assignment root
source('src/loaddata.R')
source('src/functions.R')
source('src/A1.R')
source('src/A2.R')

SAVEPLOTS = TRUE

off.diag.to.test = c(0.01, 0.05, 0.1)


lambdas.2 = list(c(2,8), c(3,7), c(4,6))
fit.2.res = calculate.fit.result.em(lambdas.2, off.diag.to.test, 2,
                                    delta=c(0.5,0.5))
# One fit chosen for the report

lambdas.3 = list(c(1,5,9), c(2,5,8), c(3,5,7))
fit.3.res = calculate.fit.result.em(lambdas.3, off.diag.to.test, 3,
                                    delta=c(1/3,1/3,1/3))
# One fit chosen for the report
fit.3 = fit.3.res[[8]]

lambdas.4 = list(c(1, 4, 6, 9), c(2, 4, 6, 8), c(1, 4, 8, 12))
fit.4.res = calculate.fit.result.em(lambdas.4, off.diag.to.test, 4,
                                    delta=c(0.25,0.25,0.25,0.25))
# One fit chosen for the report
```

A.14 Local and global decoding and state predictions

```
# This source file assumes that src/fit-em.R has
# been run.


local.decoding = pois.HMM.local_decoding(x, 3, fit.3$lambda, fit.3$gamma,
                                         fit.3$delta)

global.decoding = pois.HMM.viterbi(x, 3, fit.3$lambda, fit.3$gamma,
                                   fit.3$delta)

plot.and.save('local-decoding.pdf', 12, 7, 1.5, plot.predicted.states,
              x, local.decoding, fit.3$lambda,
              main="Local decoding for EM fitted 3-state Poisson HMM",
              xlab="Week", ylab="Weekly soap sale")

plot.and.save('global-decoding.pdf', 12, 7, 1.5, plot.predicted.states,
              x, global.decoding, fit.3$lambda,
              main="Global decoding for EM fitted 3-state Poisson HMM",
              xlab="Week", ylab="Weekly soap sale")

state.preds = pois.HMM.state_prediction(x, 3, fit.3$lambda, fit.3$gamma,
                                        H=10)
```

References

[1] Olivier Cappé, Eric Moulinen, and Tobias Rydén. *Inference in Hidden Markov Models.* Springer New York, 1st edition, 2005.

[2] Jim Pitman. *Probability.* Springer New York, 1st edition, 1993.

[3] Walter Zucchini and Iain L. MacDonald. *Hidden Markov Models for Time Series.* Chapman & Hall/CRC, 1st edition, 2009.