Convolutional PCA (and ICA)

Novel Insight Research, Tomas Ukkonen, 2022

Now that it is possible to calculate fast PCA using superresolutional numbers. It is possible to calculate convolutional PCA to find signals that are strong in data.

Let's define $\boldsymbol{x}(n) = [s_1(n), s_2(n), s_3(n), s_4(n), s_5(n) \ldots s_K(n)]^T$. Calculating this PCA does not allow delays in time, so we further define $\hat{\boldsymbol{x}}(n) = [\boldsymbol{x}(n+k), \boldsymbol{x}(n+k-1), \ldots \boldsymbol{x}(n-k)]$, to have maximum of k-time delay in signals. Signals $s_1(n)$ are superresolutional numbers with time history of given dimensions of superresolutional numbers (Now $d=7$, could be $d=31$).

TODO:

1. Write code stock market data and try to learn convolutional PCA.

2. Define room with K microphones (K dimensional PCA with time-delays k so there are input dimensions $2\,k\,K$). Create random audio sources at random locations and calculate measured signals at microphone locations. Try to solve inverse PCA to learn source signals where learnt signals are convolutions of the measured signals. $y_i(n) = \sum_k \sum_i a_i \circ s_i(n+k)$. Convolution should allow in audio echo models from the room.

RESULTS: ConvPCA cannot separate sources reliably(?), estimated signals are not sinusoids although sources are sinusoids with different frequencies and phases. ConvPCA gives some useful results with $k = [-11, +11]$. However, you still need Convolutional ICA(?). By calculating covariance matrix $C_{xx} = I$ and mean $\mu_x = 0$ you get proper results so convolutional PCA works and correlations between variables are removed.

Convolutional ICA

Convolutional ICA need to calculate. FastICA don't seem to work. Investigate what are distributions of $p(y)$: $y = w^T x$ when using superreso numbers, are they same with normal distributed numbers? Using FastICA to single number dimensions doesn't seem to work. Distributions are not same as with normal distributed numbers.

In FastICA non-linearity $G(u) = -e^{-u^2/2}$ measures non-gaussianity which maximum is at when $u \rightarrow$ large. To map this to superresolutional numbers you measure non-gaussianity per component number: $G(s) = -\prod_i G(s_i) = -e^{-0.5\sum_i s_i^2} = -e^{-0.5\|s\|_F^2}$. Now this derivate per $s = \boldsymbol{w}^T\boldsymbol{x}$ is difficult problem. We can write superresolutional vector $\boldsymbol{x}$ as matrix $\boldsymbol{X}$ where components of rows are superresolutional values for Frobenius norm.

$$\frac{\partial G(s = \boldsymbol{w}^T\boldsymbol{x})}{\partial \boldsymbol{w}} = \frac{1}{2} e^{-0.5\|s\|_F^2} \frac{\partial \|\boldsymbol{w}^T\boldsymbol{X}\|_F^2}{\partial \boldsymbol{w}}, \ \frac{\partial \|\boldsymbol{w}^T\boldsymbol{X}\|_F^2}{\partial \boldsymbol{w}} = \frac{\partial \mathrm{tr}(\boldsymbol{w}^T\boldsymbol{X}\,\boldsymbol{X}^T\boldsymbol{w})}{\partial \boldsymbol{w}} = 2\,\boldsymbol{X}\,\boldsymbol{X}^T\boldsymbol{w}$$

$$\nabla_{\boldsymbol{w}} G(\boldsymbol{w}) = \frac{\partial G(s = \boldsymbol{w}^T\boldsymbol{x})}{\partial \boldsymbol{w}} = e^{-0.5\|s\|_F^2} \boldsymbol{X}\,\boldsymbol{X}^T\boldsymbol{w}$$

And for FastICA you also need second derivate fo $G(s)$

$$H(G(\boldsymbol{w})) = \frac{\partial^2 G(s = \boldsymbol{w}^T\boldsymbol{x})}{\partial^2 \boldsymbol{w}} = -e^{-0.5\|s\|_F^2} \boldsymbol{X}\,\boldsymbol{X}^T\boldsymbol{w}\,\boldsymbol{w}^T\boldsymbol{X}\,\boldsymbol{X}^T + e^{-0.5\|s\|_F^2} \boldsymbol{X}\,\boldsymbol{X}^T$$

We don't take approximation $E_{\boldsymbol{x}}\{\boldsymbol{X}\,\boldsymbol{X}^T\} = \boldsymbol{I}$. In this case we must solve whole Hessian matrix from the data, and hope it is well-defined and calculate it's inverse.

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha\, H(\boldsymbol{w}_t)^{-1}\,\nabla G(\boldsymbol{w}_t),\ \alpha = 1$$

By using this update rule you get result where weigh vectors don't change at all(?). The dot products are 1 in FastICA algorithm so weight vectors are not updated when $\alpha = 1$. You should do line-search of α to maximimze $G(s = \boldsymbol{w}^T\boldsymbol{x})$ so to increase α until it becomes too large or $G(s)$ becomes larger.. NOW: Just try different values α to check if changing the value matters.

TODO: set time-delays to 0 so you get regular ICA problem. Try to get superreso ICA solve it correctly..

FIX(?):

Are the gradients really:

$\nabla_{\boldsymbol{w}} G(\boldsymbol{w}) = \frac{\partial G(s=\boldsymbol{w}^T\boldsymbol{x})}{\partial \boldsymbol{w}} = e^{-0.5\|s\|_F^2}\boldsymbol{x}\,(\boldsymbol{x}^T\boldsymbol{w}) = e^{-0.5\|s\|_F^2}\boldsymbol{x}\,s$

And for FastICA you also need second derivate fo $G(s)$ $(\boldsymbol{x}\,\boldsymbol{x}^{\boldsymbol{T}} = \boldsymbol{I})$

$H(G(\boldsymbol{w})) = \frac{\partial^2 G(s=\boldsymbol{w}^T\boldsymbol{x})}{\partial^2 \boldsymbol{w}} = -e^{-0.5\|s\|_F^2}s^2\,\boldsymbol{x}\,\boldsymbol{x}^T + e^{-0.5\|s\|_F^2}\boldsymbol{x}\,\boldsymbol{x}^T = e^{-0.5\|s\|_F^2}(1-s^2)\boldsymbol{I}$

This means Newton iteration simplifies to

$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \alpha\,E_{\boldsymbol{x}}\{H(\boldsymbol{w}_t)\}^{-1}E_{\boldsymbol{x}}\{\nabla G(\boldsymbol{w}_t)\} = \boldsymbol{w}_t - \alpha\,E_{\boldsymbol{x}}\{e^{-0.5\|s\|_F^2}(1-s^2)\}^{-1}E_{\boldsymbol{x}}\{e^{-0.5\|s\|_F^2}\boldsymbol{x}\,s\}$

This don't work, we don't assume $\boldsymbol{x}\,\boldsymbol{x}^T = \boldsymbol{I}$ so we get:

$H(G(\boldsymbol{w})) = \frac{\partial^2 G(s=\boldsymbol{w}^T\boldsymbol{x})}{\partial^2 \boldsymbol{w}} = e^{-0.5\|s\|_F^2}\boldsymbol{x}\,\boldsymbol{x}^T(1-s^2)$

And we need to compute for Hessian matrix..

ABOVE DON'T WORK, MAYBE WE NEED TO HAVE:

$\frac{\partial\|\boldsymbol{w}^T\boldsymbol{x}\|_F^2}{\partial \boldsymbol{w}} = \frac{\partial < \boldsymbol{w}^T\boldsymbol{x}, \boldsymbol{w}^T\boldsymbol{x} >_F}{\partial \boldsymbol{w}} = 2 < \boldsymbol{x}, \boldsymbol{w}^T\boldsymbol{x} >_F = \sum_i \boldsymbol{x}(d)_i(\boldsymbol{w}^T\boldsymbol{x})_i$

$\nabla_{\boldsymbol{w}} G(\boldsymbol{w}) = \frac{\partial G(s=\boldsymbol{w}^T\boldsymbol{x})}{\partial \boldsymbol{w}} = e^{-0.5\|s\|_F^2} < \boldsymbol{x}, \boldsymbol{w}^T\boldsymbol{x} >_F$

Now Hessian matrix for this one is:

$H(G(\boldsymbol{w})) = \frac{\partial^2 G(s=\boldsymbol{w}^T\boldsymbol{x})}{\partial^2 \boldsymbol{w}} = -e^{-0.5\|s\|_F^2} < \boldsymbol{x}, \boldsymbol{w}^T\boldsymbol{x} >_F < \boldsymbol{x}, \boldsymbol{w}^T\boldsymbol{x} >_F^T + e^{-0.5\|s\|_F^2} < \boldsymbol{x}, \boldsymbol{x}^T >_F$

PSEUDOINVERSE KOODISSA OLI VIKAA. *matrix.inv() koodi toimii sen sijaan OIKEIN superresolutional numeroilla! Nyt ICA koodi konvergoituu järkevästi johonkin, mutta mikä on konvergoituva ratkaisu???*

Compare this with Linear ICA where input data is data with added time-delays $\hat{\boldsymbol{x}}(n)$. Try to solve demixing error from a room with random mic and audio signal sources. Another problem is Brain EEG measurements where measurement points are fixed near brain and you learn deconvolutional sources from brain. With Interaxon Muse you have only 4 measurement points so you get 4 signals.

Research problem: how to do convolutional ICA, you get signals with finite time horizon $s_i(n)$ and You need to do convolution $b \circ s_i(n)$ so that you maximize non-gaussianity/kurtosis or something like that given finite time horizon (single variable $s_i(n)$) and multiple samples $\{s_i(n-k)\}_k$.

Tomas Ukkonen, M.Sc.