Chebysev method for the acceleration of CNN deconvolution algorithms.

László Orzó, Tamás Szirányi, Tamás Roska

Abstract: CNN algorithms for deconvolution is already known, and frequently applied in different deblurring and convolution/deconvolution tasks. Usually the speed of the deconvolution on a CNN chip is very high due to its analog and parallel design. The duration of the transient is $k\tau$, where $\tau$ is the time constant of the CNN (for the shake of simplicity we can take $\tau=1$) and $k$ can be as low as 4-15 for typical templates. There are, nevertheless, badly conditioned templates, when the length of the transient can be extremely high. By the linear algebraic Chebysev method we can speed up the processing of these deconvolution, template inversion algorithms. It is necessary to fit the original algorithm to the CNN constrains. We will demonstrate that the speed of convergence can be three or four times faster if we apply appropriate subsequent re-scale of the used templates.

Introduction

For a wide class of convolutions, there exist CNN algorithms, which can compute its inverse\textsuperscript{1}. That is, if we know the convolution kernel, the B template, and the image generated by it, then we can retrieve the original image by an appropriate deconvolution procedure. There can be several reasonable practical applications\textsuperscript{2,3,4,5,6}. For the deconvolution one form of the well-known Jacobi method was implemented. It is one of the simplest but robust linear algebraic iterative matrix inversion algorithm. Applying this method within the CNN paradigm we have to use an appropriately defined A template. It is a fundamental component of the CNN paradigm\textsuperscript{7}. In our case the A template\textsuperscript{1} has to be equal to 1-B (1). In the followings we shall use unit time constant and unit time step for the sake of simplicity.

The effects of a convolution, or a B template on an image can be described as a suitable matrix multiplied by the image vector

$Y = BU$

$X_{n+1} = AX_n + Y$  \text{where } A = 1 - B \hspace{1cm} (1)$

$X_\infty = U$

Here $A$ and $B$ hyper-matrices corresponding to the convolutions of $A$ and $B$ templates respectively. $U$ is the original image, $Y$ denotes the result of the convolution on it. Finally $X_\infty$ indicates the reconstructed original image.
It happens frequently that the matrix, and so its generator template, is badly conditioned. That is the biggest and smallest eigenvalues' ratio is very high. Applying the Jacobi method we can assure only a very slow convergence in this cases. For any type of practical application it is indispensable to enhance the speed of the otherwise slow algorithm.

There are several linear algebraic methods that lead to the speeding up. These methods usually use as parameters: the minimal ($\alpha$) and maximal ($\beta$) eigenvalues of the matrix. These parameters can be easily determined, because this matrix is derived from a convolution. It is well known, that the products of the template’s Fourier transform, which is otherwise real, and the Fourier transform of the original image is equal to the Fourier transform of the convolved image. The Fourier transformation constructs a diagonal representation of the matrix and so from the templates Fourier transform all the eigenvalues can be determined.

In the majority of acceleration methods the next type of iteration is used:

$$X_{n+1} = X_n - \tau_i(BX_n - Y)$$

(2)

Here $\tau_i$ is a parameter, which appropriate definition direct to the enhancement of the convergence’s speed. In the Jacobi formula $\tau_i$ has to be chosen smaller than $2/\beta$ to ensure convergence. In the case of so called ‘Shift Technique’ $\tau_i$ is set to be $2/(\alpha+\beta)$. By this criterion the convergence speed can be duplicated. We can reach further increase in convergence speed if we use a ‘Non-Stationer Iterative Procedure’. In this procedures $\tau_i$ parameter changes in every step of the iteration. These techniques seem to be adequate for application, because downloading of a template is very fast in the CNN universal machine.

The fastest general convergence can be achieved by the Chebysev method. (Even the above mentioned ‘Shift Technique’ is a particular case of it.)

To apply this algorithm it is required that the iteration reaches the best approximation of the perfect solution in the $j^{th}$ step. The evolving error term can be characterized by a same type of iteration as the state variable or image themselves. So the iteration on the error term can be interpreted as an increasing rank matrix polynomial of the original matrix ($B$ template) effects on the initial error term. We can expand this equation appropriately to the matrix’s eigenvectors, and so we can express evolving error term components as a scalar polynomial of the eigenvalues multiplied by the initial error appropriate component.

$$\xi_j = \prod_{i=1}^{j}(E - \tau_iB)\xi_0$$

$$\xi_j^n = P_j(\lambda^n)\xi_0^n$$

The final error is minimized iff this polynomial approximates zero for all the eigenvalues. The polynomial which provides the best approximation of the identically zero function in the [-1,1] interval is the Chebysev polynomial.

$$T_j(x) = \cos(j \cdot \arccos(x))$$

(4)
By an appropriate transformation we can map the $[\alpha, \beta]$ interval to [-1,1]. Such a way the optimal $\tau_i$ parameters can be defined by the next way:

$$X_{n+1} = (1 - \tau_i)X_n - \tau_i((1 - B)X_n - Y)$$

$$\tau_i = \frac{2}{\beta + \alpha - (\beta - \alpha)\cos(\frac{2i-1}{2j}\pi)} \quad (i = 1...j) \quad (5)$$

We can see from the equation (5), that it is satisfactory if we change the time step appropriately in CNN simulators.

**Examples**

Here we will show two examples for the application of the above method in the CNN.

First we use a badly conditioned template:

$$B = \begin{bmatrix}
-0.05 & -2 & -0.05 \\
-2 & 1.05 & -2 \\
-0.05 & -2 & -0.05 \\
\end{bmatrix} \quad (6)$$

The Jacobi method converges very slowly. Figure 1 shows the original test image (Figure 1. A) and the effects of a B template (6) on it (Figure 1. B). On Figure 2 we can see the results of the hitherto used inversion algorithm after several steps of iteration. After thirty steps of iteration the root mean square error was still: **RMSE = 0.085472**

![Figure 1](image1.png)  ![Figure 2](image2.png)

Figure 1. Here we can see the original test image (A) and the effects of a B (6) template on it (B).
Figure 2. We can see the effects of the Jacobi method after 10 (A), 20 (B) and 30 (C) steps of iteration. The subplots (D, E, F) show their (A, B, C) differences from the original image (Figure 1. A) respectively.

We can define the maximal ($\beta$) and the minimal ($\alpha$) eigenvalues from the template two dimensional Fourier transform (Figure 3.). $\beta$=1.6495; $\alpha$=.05;

![Spectrum of a 3x3 CNN template](image)

Figure 3. The two dimensional Fourier transform of the (7) B template.

We have got the next $\tau_i$ parameter values for $j=10$ (6).

| $\tau_i$ parameter values in decreasing order for ($j=10$; $\beta=1.65$; $\alpha=0.05$) |
|------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| 16.7086          | 7.2889          | 3.5172          | 2.0542          | 1.3796          | 1.0255          | 0.8243          | 0.7064          | 0.6399          |

Table 1
We can see that for small $i$, the $\tau_i$ parameter values are extremely high. These high values can force the CNN cells to the non-linear range of dynamics. Therefore we used the $\tau_i$ parameters in increasing order (or with decreasing indices) and after finishing it we run the Jacobi method ($\tau_i = 1$) for five more steps to eliminate the appearing saturation caused noise.

Instead of downloading in every step of the iteration a new re-scaled template, defined by the appropriate parameters, in the CNN simulator we can simply adjust the time step parameters adequately. It leads to an identical iteration, because the simulators use the Euler approximation of the CNN cell’s differential equation (5). In this case after fifteen steps of the processing (Figure 4) the root mean square error of the retrieved image is:

$$\text{RMSE} = 0.024671$$

![Figure 4. The result of the fifteen steps of Chebysev type inversion method (A) and its difference (B) from the original image.](image)

It can be seen that by the Chebysev method in half amount of steps we can reach far better approximation of the original image than by the Jacobi process. For the implementation of the Chebysev method to the CNN it is necessary to avoid the appearance of extremely high parameter values. It can be done by the application of shorter sequence of Chebysev parameters and iterating it several times. It will decrease the gain, but prevents the artifacts.

On Figure 5 we can compare the evaluation of the RMSE during the processing of different deconvolution, template inversion algorithms. We can see here the speed of the convergence in the original (Jacobi) method, the ‘Shift’ technique, and two Chebysev like algorithms, within which the four-step Chebysev ($j=4, \beta=1.6495, \alpha=0.05$) and the eight-step Chebysev ($j=8, \beta=1.6495, \alpha=0.05$) was iterated.
Figure 5. We can compare the speed of the convergence of different deconvolution algorithms.

In the second example we examine the deblurring algorithms in the case of a singular (non-invertable) template:

\[
B = \begin{bmatrix}
0.11 & 0.11 & 0.11 \\
0.11 & 0.11 & 0.11 \\
0.11 & 0.11 & 0.11 \\
\end{bmatrix}
\]  

(7)

This template is blurring the original test image (Figure 6).

Figure 6. Here can be seen the results (B) of the blurring template (7) on a (A) test image.

If the matrix, implicated by the convolution, has a zero eigenvalue, then it can not be inverted perfectly. In this case there is some part of the original image, which lays in the direction of the zero eigenvalue’s eigenvector. Loosing all the amplitude
information of this component, we will not be able to regain it during the inversion. Nonetheless, this type of error can be relatively small in the case of natural images. This property can appear frequently. For example, if the matrix has both negative and positive eigenvalues, then due to the origin of the matrix is a small support convolution, the eigenvalues follow continuous distribution. So there has to be zero eigenvalue as well. The second case when the template’s effect can not be inverted, when the dynamics of the cells reach the nonlinear range of the CNN. It will also lead to the loss of information.

In our case (7) the matrix, is not positive definite, so we have to use Gauss transformation\(^8\). This transformation ensures that the matrix we will use in the iteration will be at least positive semi-definite. We can accomplish this algorithm on the CNN in two steps. Such a way we can attain convergence, but sometimes a very slow one. We have to use the next procedure:

\[
\begin{align*}
X_{n+1} &= X_n - BX_{n,\alpha} \\
X_{n,\alpha} &= BX_n - Y
\end{align*}
\]  

(8)

From the templates two-dimensional Fourier transform (Figure 7) we can see that it has negative and positive eigenvalues (and so zero eigenvalue) as well: \(\beta = 0.99; \alpha = -0.3292;\)

![Spectrum of a 3x3 CNN template](image)

Figure 7. The two dimensional Fourier transform of the B (7) template.

From these data we can conclude that the exact inverse can not be achieved, but we can try to give a fairly good approximation. We have made a comparison of the Jacobi and the Chebysev methods, using the above mentioned Gauss transformation for the inversion. In both cases we have run the algorithms for 20 steps and tested which technique produce smaller root mean square error. Here we replaced the \(\alpha\) parameter by a small constant (0.1), instead of using the exact value that would be zero. By this way we can encompass the appearance of extremely high \(\tau_i\) parameter
values. We used here an other trick to avoid the nonlinear range of the cell’s dynamics. That type of noise, which is appearing due to the saturation, has a zero eigenvector component as well. This component can not be eradicated by the succeeding steps of the algorithm. Therefore we divided the input by two and after terminating the iteration we re-scaled (multiplied by two) the result. So it has greater probability, that we can keep the system dynamics within the linear range. The effectiveness of this algorithm will expectedly decrease comparing to the previous example, as we could not use the algorithm with perfect parameters. The used $\tau_i$ parameters determined by $j=20$, $\beta=.98$, $\alpha=0.1$ can be seen in Table 2.

<table>
<thead>
<tr>
<th>$\tau_i$ parameter values in decreasing order ($j=20$; $\beta=0.98$; $\alpha=0.1$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.7406</td>
</tr>
</tbody>
</table>

Table 2.

We can see the results of the inversion algorithms (Figure 8.) using the Jacobi and the Chebysev techniques. The root mean square errors were:

$$\text{RMSE(Jacobi)} = 0.09227 \quad \text{RMSE(Chebysev)} = 0.069312$$

Figure 8. We can see the results of the ‘two-step’ method of template inversion after 20 step of iteration using the Jacobi (A) and the Chebysev (B) technique. (C) and (D) shows the differences of the results from the original image (Figure 6.A).
If we use an alternative Chebysev like algorithm, the $\tau_i$ parameters are determined by $j=4$, $\beta=.99$, $\alpha=0.05$. We repeated it five times and so we reached an even lower root mean square error value. **RMSE(Chebysev') = 0.061326**

On the last figure (Figure 9.) we can see the effect of this last type of algorithm on the speed of the convergence comparing to the original one. For smaller $\alpha$ values we get higher $\tau_1$ parameters and it will saturate some of the CNN cells. It can cause the transient increase of the root mean square error (for example for $\alpha=.05$, not shown) even if it leads to a better approximation later.

![Speed of convergence](image)

Figure 9. We can compare the speed of the convergence of the inversion method. The applied version of the Chebysev technique ($j=4, \beta=.99, \alpha=.1$) converges must faster for both test pictures (Mandrill and Clown).

**Conclusion**

We demonstrated that applying the Chebysev method we can accelerate considerably the convergence of the CNN deconvolution algorithms. Even in the case of non-invertable templates we can reach a relatively accurate solution. There are other linear algebraic algorithms which can increase the speed of the convergence, like the Iterative Gradient method or Smallest Error Term's technique. However, in these cases we need to compute functionals which can be hard to determine by CNN. (For example: The overall sum of the image pixel values.)

Further scrutinized investigations seems to be necessary to define the optimal parameter values, regarding the CNN cells' non-linearity and their analog type of computation.
References


3 T. Szirányi, L. Nemes, (1995 );"Robustness of Cellular Neural Networks in Image Deconvolution and Texture Segmentation",Research report of the Analogic (Dual) and Neural Computing Systems Laboratory, (DNS-7-1995), Budapest, MTA SZTAKI


