DeconvTest: Simulation framework for quantifying errors and selecting optimal parameters of image deconvolution

Anna Medyukhina1,3 | Marc Thilo Figge1,2*

1Applied Systems Biology, Leibniz Institute for Natural Product Research and Infection Biology, Hans-Knöll-Institute, Jena, Germany
2Institute of Microbiology, Faculty of Biological Sciences, Friedrich-Schiller-University Jena, Jena, Germany
3Center for Bioimage Informatics, St. Jude Children’s Research Hospital, Memphis, TN

*Correspondence
Marc Thilo Figge, Applied Systems Biology, Leibniz Institute for Natural Product Research and Infection Biology, Hans-Knöll-Institute, Adolf-Reichwein-Strasse 23, 07745 Jena, Germany.
Email: thilo.figge@leibniz-hki.de

Present address
Anna Medyukhina, Center for Bioimage Informatics, St. Jude Children’s Research Hospital, Memphis, TN

Abstract
Deconvolution is an essential step of image processing that aims to compensate for the image blur caused by the microscope's point spread function. With many existing deconvolution methods, it is challenging to choose the method and its parameters most appropriate for particular image data at hand. To facilitate this task, we developed DeconvTest: an open-source Python-based framework for generating synthetic microscopy images, deconvolving them with different algorithms, and quantifying reconstruction errors. In contrast to existing software, DeconvTest combines all components required to analyze deconvolution performance in a systematic, high-throughput and quantitative manner. We demonstrate the power of the framework by using it to identify the optimal deconvolution settings for synthetic and real image data. Based on this, we provide a guideline for (a) choosing optimal values of deconvolution parameters for image data at hand and (b) optimizing imaging conditions for best results in combination with subsequent image deconvolution.

KEYWORDS
deconvolution, open-source software, performance evaluation

1 INTRODUCTION

Microscopy imaging has long become a powerful tool not just to visualize, but also quantify biological samples—thanks to the automated analysis of image data. Before analyzing the images, one usually has to deconvolve them in order to correct image blur caused by the microscope's point spread function (PSF). This procedure is particularly relevant for such techniques as two-photon microscopy, where the PSF contribution is especially big. Despite the number of existing deconvolution methods [1], the choice of the most appropriate algorithm and its settings remains a challenge. Most of the algorithms have multiple parameters to adjust, but there is no clear guideline for how to choose their values depending on the properties of available image data. In practice, the parameter values are often manually fine-tuned based on just looking at a few deconvolved images, whereas the remaining images get poorly reconstructed and quantified.

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.
© 2020 The Authors. Journal of Biophotonics published by WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim
To optimize the image reconstruction, we need to objectively quantify the errors produced by a particular deconvolution algorithm and its specific parameter set. While previous studies compared some deconvolution software [2, 3], they did not analyze the numerous parameters of individual algorithms. It is also not clear how the performance of the tested software will change when applied to images with different PSF contributions, voxel sizes or noise levels. To investigate the effects of these factors, we need tools for generating synthetic ground truth data, accessing deconvolution parameters, and quantifying reconstruction errors in a high-throughput mode. To conveniently provide these tools to a random image analyst, they have to be combined in a single well documented and open-source software.

Many software packages include some of these tools, but none of them has the functionality needed to quantify deconvolution performance in a systematic way (Table 1). Thus, many packages, such as CytoPacq, [4, 5], provide synthetic data by generating so-called digital phantoms of cells or tissues [6]. Usually, these software tools account for the blur, noise and finite voxel size introduced by the microscope and the image detector, which makes the resulting images realistic and extremely helpful for the validation and benchmarking of image analysis algorithms. Other software packages provide tools related to deconvolution. Thus, the PSF generator [7] generates and visualizes PSFs of various shapes, whereas the Fiji plugin “Iterative Deconvolve 3D” [8] and similar software can reconstruct the given image with a specific deconvolution algorithm. A comparison of multiple algorithms within one software is possible with the Fiji plugin DeconvolutionLab2 [9]. It provides a user-friendly interface, tools to generate digital phantom images and PSFs, as well as multiple algorithms to deconvolve image data. Unfortunately, this plugin does not explicitly quantify deconvolution errors, but only computes the signal-to-noise ratio (SNR) and other properties of the input and output images. The available batch mode involves a manual adding of selected images to a job list, which prevents a high-throughput analysis.

Thus, none of the available software allows the user to generate ground truth data, simulate microscopy imaging, run multiple deconvolution methods, quantify reconstruction errors and test multiple parameters of these steps in a single simulation workflow. To address this challenge, we created DeconvTest: a Python-based parallel simulation framework, which provides all tools needed to efficiently and systematically analyze deconvolution performance achieved with various algorithm parameters and experimental settings. DeconvTest also allows the user to compare various imaging conditions and to adjust and select them in view of optimal
results for the combined process of imaging and deconvolution.

2 | METHODS

The DeconvTest framework is available on GitHub as open-source software (https://github.com/applied-systems-biology/DeconvTest) with detailed online documentation (https://applied-systems-biology.github.io/DeconvTest/). The framework provides tools to generate synthetic microscopy data, deconvolve images and quantify reconstruction errors and thus enables basic quantitative deconvolution experiments. These in silico experiments are specified via a configuration file and executed from the command line, whereas individual modules and functions of the framework can be accessed from the Python environment. Due to the modular structure of the framework, its basic functionality can be easily extended by adding new synthetic objects, deconvolution methods and performance measures. The public availability of the framework makes it possible to implement new features not only in our own future work but also with the help of the GitHub community.

The framework consists of three main modules: in silico microscopy, deconvolution and performance quantification (Figure 1). The details of the modules are described in the following sections.

2.1 | In silico microscopy module

The goal of the in silico microscopy module is to generate synthetic image data for testing and comparing different deconvolution methods. The module simulates microscopy imaging in three steps: it convolves the digital phantom image with a PSF, downsamples the convolved image and adds synthetic noise (Figure 1). The convolution step starts with discrete images of high resolution (see Supplementary Methods for details), which approximates the continuous convolution of the sample with the PSF of the microscope. The subsequent downsampling and noise addition steps simulate the process of image discretization by recording it on a detector, such as a CCD camera. This three-step process is similar to how realistic microscopy images are generated in CytoPacq [4, 5] but different from the DeconvolutionLab2 plugin [9], which neglects the effects of discretization by skipping the downsampling step (Table 1).

The digital phantoms available in DeconvTest have either ellipsoidal or spiky cell shape (see Supplementary Methods for details). The objects with ellipsoidal shape are produced by thresholding of a 3D Gaussian kernel, while more complex shapes are achieved by generating spikes of various amplitudes and smoothness on the ellipsoid surface. The modular organization of the framework allows integrating other types of digital phantoms, for example, those provided by CytoPacq and other software for synthetic image generation [2, 4, 5, 9].

The generated digital phantoms are convolved with the PSF images, which are generated as 3D Gaussian kernels. The size of the PSF is defined by two parameters: the PSF width (SD) in \( \text{xy} \) and the PSF aspect ratio, which is the ratio between the PSF width in \( z \) and \( \text{xy} \). The Gaussian kernel adequately approximates the PSF shape in multiphoton microscopy [10], but is rather far from the PSF shape in confocal microscopy and other techniques. Other PSF shapes [7, 9] will be implemented in future extensions, whereas the current version of the framework focuses on simulating images realistic for multiphoton microscopy.

After convolving with a PSF, images are downsampled to a specified voxel size and corrupted by synthetic Poisson or Gaussian noise. The downsampling occurs by linear interpolation, and the target voxel size can be either isotropic (equal values for \( x, y \) and \( z \)), or anisotropic (different values for \( z \) and \( xy \)). Poisson and Gaussian noise of a specified SNR (see Supplementary Methods) can be added in arbitrary order, and the sequence of the downsampling and noise addition steps can also be specified arbitrarily.

2.2 | Deconvolution module

Deconvolution module reconstructs the generated synthetic images with one of the integrated algorithms. It
provides a single unified interface for several deconvolution plugins initially implemented in Fiji [11]. The so-far integrated plugins include “DeconvolutionLab2” (http://bigwww.epfl.ch/deconvolution/deconvolutionlab2/) [9] and “Iterative Deconvolve 3D” (https://imagej.net/Iterative_Deconvolve_3D) [8].

DeconvolutionLab2 includes multiple algorithms, of which we so far integrated the most commonly used ones: the regularized inverse filter (RIF) and the Richardson–Lucy with total variance (RLTV). In both algorithms, one has to specify the regularization parameter $\lambda$, which is used to penalize noise amplification (Table 2). Furthermore, the RLTV algorithm—being an iterative minimization approach—also accepts the number of iterations as input and thus has two adjustable parameters (Table 2).

The plugin “Iterative Deconvolve 3D” uses an improved version of the deconvolution approach for the mapping of acoustic sources (DAMAS) algorithm [8]. DAMAS is an iterative nonnegative least squares solver, which includes regularization by a low pass filter and Wiener filter. The number of algorithm iterations is specified via the termination parameter $\Delta$ by stopping the iteration if the changes in the image are less than the value of this parameter (Table 2). Smaller values of the termination parameter correspond to more iterations of the algorithm, whereas with higher values of $\Delta$, DAMAS will converge sooner. The meaning of these and other parameters of DAMAS, RIF and RLTV are summarized in Table 2.

The integrated plugins are executed from Python and thus can be easily incorporated in the DeconvTest analysis pipeline. The values of the deconvolution parameters, as well as the paths to the input file, PSF and the output file are passed to Fiji using the headless mode. The unified Python interface allows the user to run deconvolution in a parallel manner, record the run-time and test different combinations of the plugins’ input parameters.

Though only three algorithms from two Fiji plugins are available for use so far, any other deconvolution software can be integrated without the need of re-implementing the corresponding algorithm, as long as the software can run from the command line. Thus, in contrast to DeconvolutionLab2, which relies on internally implemented algorithms, DeconvTest allows comparing deconvolution methods implemented within different software packages.

2.3 | Performance quantification module

The performance quantification module compares the deconvolved images to the ground truth (digital phantoms) and quantifies the reconstruction errors. Before comparing the images, the deconvolved images are upsampled to match the voxel size of the ground truth. The deconvolution errors are quantified by computing the normalized root-mean-square error (NRMSE):

$$NRMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - x_i)^2}{N \left(x_{\text{max}} - x_{\text{min}}\right)}}$$

Here, $x_i$ and $y_i$ are the $i$th pixel’s intensities in the ground truth and the deconvolved image respectively, $N$ is the total number of pixels in the ground truth image, and $x_{\text{max}}$ and $x_{\text{min}}$ ($x_{\text{max}} > x_{\text{min}}$), are the maximum and the minimum intensity values of the ground truth image.

The computed performance values are saved in a csv file, where each line corresponds to an individual image.

### Table 2: Parameters of integrated deconvolution algorithms

| Algorithm | Parameter | Type   | Parameter meaning                                      |
|-----------|-----------|--------|--------------------------------------------------------|
| RIF       | $\lambda$ | Float  | Regularization parameter to penalize noise amplification |
| RLTV      | $\lambda$ | Float  | Regularization parameter to penalize noise amplification |
|           | Iterations| Integer| Number of iterations for the Richardson–Lucy algorithm |
| DAMAS     | Normalize | Boolean| Normalizes the PSF before deconvolution                |
|           | Perform   | Boolean| Performs anti-ringing step, which reduces artifacts from features close to the image border |
|           | Detect    | Boolean| Detects divergence: stops the iteration if the changes in the image increase |
|           | Wiener    | Float  | Size of the Wiener filter $\gamma$, regularization parameter |
|           | Low       | Integer| Size of the low pass filter in pixels (in $x$, $y$ and $z$), regularization parameter |
|           | Terminate ($\Delta$) | Float | Terminates iteration if the changes in the image are less than the value of this parameter |
|           | Iterations| Integer| Maximum number of iterations of the algorithm |

Abbreviations: RIF, regularized inverse filter; RLTV, Richardson–Lucy with total variance; DAMAS, deconvolution approach for the mapping of acoustic sources.
deconvolved with a specific algorithm and specific parameter values. This format makes it convenient to analyze and plot the results both within Python and using other software tools.

2.4 Setting up in silico experiments

Each in silico experiment is described by a configuration file and executed from the command line. The configuration file specifies the simulation steps, types and sizes of digital phantoms and PSFs, image voxel sizes, noise levels, as well as deconvolution algorithms and their parameters. Multiple values of all these can be provided and tested in a single simulation run, which enables a systematic study of the given parameters and their effect on deconvolution errors. If desired, the users can apply parts of the DeconvTest workflow to their own synthetic data or deconvolved images by specifying corresponding simulation steps in the configuration file. Each DeconvTest simulation is executed as a specified number of parallel processes, which allows the user to exploit the power of computational servers and supercomputers.

3 RESULTS AND DISCUSSION

To showcase the use of our framework, we explored how reconstruction performance depends on the properties of image data and the values of deconvolution parameters. We start with identifying the optimal parameters of the integrated deconvolution algorithms and comparing the performance of the algorithms relative to each other. We continue by analyzing how deconvolution performance depends on the properties of image data, such as voxel size, noise level and the size of the microscope’s PSF. We next investigate how the PSF size and noise level determine the deconvolution settings that should be used for optimal image reconstruction. Finally, we demonstrate how our framework can help to optimally deconvolve real microscopy images.

3.1 DeconvTest helps to identify the optimal deconvolution settings

First, we applied the DeconvTest framework to identify the parameters of the integrated deconvolution algorithms that result in the lowest reconstruction errors (NRMSE) and the lowest computation time. We generated synthetic data with varying PSF sizes, voxel sizes and noise levels, which resulted in a total of 40 images (Table 3: Workflow 1). We then deconvolved the images using different combinations of algorithm settings and evaluated the reconstruction errors and computation times for each of them. This procedure allowed us to identify the optimal values of deconvolution parameters, compare between different methods, and explore which parameters and in what way affect the algorithm performance (Figures 2, S1).

We started with investigating the parameters of the RIF and RLTV algorithms, as they have the fewest parameters to adjust. For the single parameter $\lambda$ of the RIF algorithm, the lowest NRMSE was achieved for the value $\lambda = 100$ (Figure 2A), whereas the related RLTV parameter produced the lowest errors with the value $\lambda = 0.01$ (Figure 2C). The computation time of both methods did not depend on $\lambda$ (Figure 2B,D), but increased with the number of RLTV iterations (Figure 2D). Surprisingly, applying more iterations of RLTV did not improve the quality of image reconstruction: as few as five iterations of this algorithm produced the most accurate result (Figure 2C).

Among the many parameters of the DAMAS algorithm, some parameters affected deconvolution performance, while others did not. Thus, reconstruction accuracy improved when applying Wiener filter or low pass filter, whereby the exact size of the Wiener filter did not matter (Figure S1c), but the size of the low pass filter had its optimum around two pixels (Figure S1a). In contrast, deconvolution errors did not depend on the number of iterations (specified by the termination parameter), performing an anti-ringing step, PSF normalization or detecting divergence (Figures 2E, S1a,c). PSF normalization, detecting divergence, and the size of the Wiener filter also did not change computation time (Figure S1b,d). The computation time was only slightly affected by the anti-ringing step (Figure 2F) and the size of the low pass filter (Figure S1b), but greatly increased with the number of iterations specified by the termination parameter $\Delta$ (Figure 2F). This diverse behavior of DAMAS parameters emphasizes the value of knowing which parameters influence deconvolution performance and which do not: such knowledge can immensely facilitate the task of adjusting algorithm settings, especially for those methods that—like DAMAS—have many parameters to specify.

To identify the exact values of the optimal parameters, we averaged the NRMSE across all 40 processed images for each parameter combination. For RIF and RLTV, we selected the parameter set that resulted in the lowest average NRMSE, because these were also the parameters that required the least computation time. For DAMAS, we had to find a trade-off between computation time and deconvolution errors. We selected the parameter combination that produced the lowest average NRMSE, but we changed the value of the termination parameter $\Delta$ from 0.001 to 0.1. This
### Table 3  Parameters of the DeconvTest workflow

| Parameter ID | Parameter | Workflow 1 Optimal settings of deconvolution algorithms (Section 3.1; Figures 2, S1) | Workflow 2 Influence of PSF and voxel size (Section 3.2; Figure 3A-D, S2a-h) | Workflow 3 Influence of noise (Section 3.2; Figure 3E,F, S2i-l) | Workflow 4 Influence of PSF size and noise on optimal settings of deconvolution (Section 3.3; Figure 4, S3-S6) |
|--------------|-----------|-----------------------------------------------------------------------------------|-------------------------------------------------------------------------------|------------------------------------------------------------------|------------------------------------------------------------------|
| p1           | Cell size, μm | 10 ± 2                              | 10 ± 2                                                                         | 10 ± 2                                                              | 10 ± 2                                                              |
| p2           | Number of unique input cells | 5                                  | 10                                                                             | 10                                                                | 10                                                                |
| p3           | PSF size, xy  | 0.5, 1.2                            | 0.5, 0.8, 1.0, 1.2                                                             | 0.5, 1.2                                                          | 0.5, 1.2                                                          |
| p4           | PSF aspect ratio | 4.5                                | 2, 3, 4, 5, 6                                                                  | 4.5                                                               | 3, 5                                                               |
| p5           | Initial voxel size, μm\[^a\] | 0.1                                | 0.1                                                                            | 0.1                                                              | 0.1                                                               |
| p6           | Voxel size after resizing, μm (z/xy) | 0.3, 0.3, 0.3               | 0.2, 0.3, 0.4, 0.5, 1/0.3, 1/0.5, 3/0.3, 3/0.5                               | 0.3, 0.3, 0.3                                                      | 1/0.3                                                             |
| p7           | Poisson noise SNR | 2, no noise                        | no noise                                                                       | 2, 5, 10, 20, 50, 100, no noise                                   | 2, 5, no noise                                                     |
| p8           | Number of parameter combinations for in silico microscopy | 8                                | 160                                                                         | 28                                                               | 12                                                               |
| p9           | Total number of cells to deconvolve (p2 × p8) | 40                              | 1600                                                                       | 280                                                              | 120                                                              |
| p10          | RIF λ       | 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10 000       | 100                                                                          | 100                                                              | 0.01, 0.1, 1, 10, 100, 1000                                        |
| p11          | RLTV λ      | 0.001, 0.01, 0.1, 5, 10, 100, 1000, 10 000 | 0.01                                                                         | 0.01                                                             | 0.001, 0.01, 0.05, 0.1                                             |
| p12          | Iterations  | 5, 10, 20, 50, 100                   | 5                                                                            | 5                                                                | 5                                                                |
| p13          | DAMAS Normalize | True, False                     | True                                                                          | True                                                              | True                                                              |
| p14          | Perform     | True, False                         | True                                                                          | True                                                              | True                                                              |
| p15          | Detect      | True, False                         | False                                                                         | False                                                             | False                                                             |
| p16          | Wiener      | 0, 0.1, 1, 10                       | 1.0                                                                          | 1.0                                                               | 0, 0.1, 1, 10                                                     |
| p17          | Low         | 0, 1, 2, 3, 4, 5                    | 2                                                                            | 2                                                                 | 0, 2, 4, 6                                                       |
| p18          | Terminate (Δ) | 0.0001, 0.001, 0.1, 1               | 0.1                                                                          | 0.1                                                               | 0.1                                                               |
| p19          | Maximum iterations | 200                              | 200                                                                          | 200                                                              | 200                                                              |
| p20          | Number of deconvolution parameter combinations | 816                         | 3                                                                            | 3                                                                 | 26                                                                |
| p21          | Total number of deconvolutions (p9 × p20) | 32 640                        | 4800                                                                         | 840                                                              | 3120                                                              |
| p22          | Total computation time, hours\[^b\] | 61                               | 2                                                                            | 0.5                                                               | 1.3                                                               |

\[^a\]See Supplementary Methods for how the voxel size was selected.

\[^b\]The computation was carried out using 75 cores of our GNU/Linux Server (SUSE) with an Intel Xeon CPU E78890 v3 (2.50 GHz) and 1514 GB RAM.
adjustment marginally increased the average NRMSE (1%) but reduced the processing time by a factor of 3.5 (Figure 2F).

Detecting the optimal parameter values allowed us to analyze how the three deconvolution algorithms perform relative to each other. We found that DAMAS...
resulted in the highest accuracy, whereas RIF—the simplest deconvolution method—was the least accurate (Figure 2G). Better reconstruction results came at the price of using more computational resources: the required processing time was the highest for the most accurate DAMAS and the lowest for the least accurate RIF (Figure 2H).

By computing NRMSE for each parameter combination, DeconvTest allowed us to make an informed and objective choice of deconvolution settings. Such choice is hardly possible when looking at a few deconvolved images, because the results might look similar for different parameter values and simultaneously vary between analyzed images (Figures 2I, S2), or...
between image properties, such as the noise level (Figure 2J).

### 3.2 Deconvolution performance depends on PSF size, image voxel size and noise level

Next, we used the DeconvTest framework and the selected optimal deconvolution settings to investigate how reconstruction performance depends on the properties of image data, such as the size of the applied PSF, image voxel size and noise level. With this analysis, we aimed to identify the optimal range of experimental settings and reveal the experimental parameters that strongly contribute to image corruption.

The analysis was done in two steps. We first generated noise-free images to study the influence of PSF size and voxel size (Table 3: Workflow 2), and then examined various noise levels for selected PSF and voxel sizes (Table 3: Workflow 3).

PSF size and voxel size affected both deconvolution errors and computation time but to a different extent. Larger PSFs caused higher deconvolution errors since they distorted the images to a higher degree (Figure S3), whereby the errors increased with both PSF $xy$ width and PSF aspect ratio (Figures 3A, S4a,e). Larger PSFs also required more processing time, especially for RLTV and DAMAS (Figures 3B, S4f). Voxel size affected computation time and deconvolution errors mainly in the extreme value range. Thus, computation time increased for isotropic voxel sizes under 0.4 $\mu$m (Figures 3D, S4d,h), while deconvolution errors were very high for a $z$ voxel size of 3 $\mu$m (Figures 3C, S4c,g). Both deconvolution errors and computation time stayed relatively constant between the voxel size values of 0.4 $\mu$m (isotropic) and 1 $\mu$m (in $z$). We, therefore, advise to use voxel sizes from this range to maximize the quality of deconvolution but keep processing time low.

Interestingly, noise affected deconvolution errors and computation time to a much lesser extent than expected. The time consumed by deconvolution did not depend on the noise SNR (Figures 3F, S4j,l). The accuracy was uniform for SNR values of 10 and higher and was undermined only by extremely high noise levels uncommon for real microscopy images (Figures 3E, S4i,k). The SNR dependence was lost entirely in DAMAS and RIFT when applying a large PSF (Figures 3E, S4i, upper curves): the errors for noise-free images were as high as for heavily noise-corrupted ones. These results suggest that the PSF,

**FIGURE 4** Deconvolution results of the DAMAS algorithm for different values of the low pass filter. A, Normalized root-mean-square error (NRMSE) depending on the noise level and the size of the point spread function (PSF). The dotted rectangles indicate the parameter values resulting in the lowest average NRMSE. The number of data points in each box $n = 40$ (10 unique input images $\times$ 4 different sizes of the Wiener filter parameter). For other PSF SDs and aspect ratios see Figure S5; for the DAMAS’s Wiener filter parameter see Figure S6; for the RIFT and RLTV $\lambda$ see Figures S7 and S8. B, Examples of deconvolved images (middle $xz$ slice) for PSF $xy$ SD 0.5 $\mu$m, aspect ratio 3, noise SNR 2 (panel A, bottom left)
rather than noise, is often the main contributor to image distortion and low deconvolution quality.

### 3.3 Optimal deconvolution parameters change depending on the properties of image data

Since reconstruction errors heavily depend on the image data properties, we were wondering whether they can be reduced by choosing the right deconvolution setting for each particular PSF size and noise level. To explore this possibility, we used DeconvTest to generate images with varying PSF sizes and Poisson noise SNRs and to reconstruct them with the three integrated algorithms, while varying some of the algorithm parameters (Table 3: Workflow 4). We then examined which values of these parameters resulted in the lowest NRMSE for specific PSF sizes and noise SNRs (Figure 4A, S5–S8).

Remarkably, for all studied parameters, the optimal parameter value increased with an increasing level of noise. This trend was observed for RIF and RLTV $\lambda$, as well as for DAMAS’s low pass filter and Wiener filter, and was especially prominent when applying small PSFs (Figure 4A, left column; Figure S5–S8, top row). All examined parameters behave similarly because they are all involved in regularization and noise filtering, and higher noise levels naturally require larger filter sizes for the optimal result.

In contrast to the noise, the effects of PSF size differed between algorithms. In most cases, large PSFs required smaller values of the regularization parameters, whereas small PSFs needed larger filter sizes. This trend was especially prominent for the low pass filter in noisy images (Figure 4A, middle and bottom rows; Figure S5, middle and right columns) and for the Wiener filter (Figure S6), but was reversed for the RIF algorithm (Figure S7). The fact that smaller values of regularization parameters mostly worked...
better for large PSFs suggests that the PSF itself may act as a regularizer by smoothing part of the image noise.

These results demonstrate that the optimal sizes of regularization filters depend on whether the noise or the PSF is the main contributor to image distortion. As a general trend observed in most cases, larger sizes of regularization filters worked better for noisy images generated with small PSFs, whereas smaller filter values were optimal for noise-free images generated with large PSFs (Figure 4A, S5, S6, S8, top right vs bottom left). This fact emphasizes the importance of choosing deconvolution settings according to the properties of image data. Such choice is not obvious from a visual inspection of deconvolved images (Figure 4B) and can be greatly facilitated by the quantitative tools available within the DeconvTest framework.

3.4 | Deconvolving microscopy images with the help of DeconvTest

Finally, we demonstrate how DeconvTest can be used to deconvolve real microscopy images. A detailed guide for this procedure is provided in the online documentation (https://applied-systems-biology.github.io/DeconvTest/, https://github.com/applied-systems-biology/DeconvTest/blob/master/docs/DeconvTest_guide_for_microscopy_image_deconvolution.ipynb), whereas here we briefly outline the steps that the users need to undertake to optimally deconvolve their images. As an example, we use a small image volume from intravital two-photon microscopy imaging of murine ear skin (Figure 5A) [12].

First, the user needs to identify the properties of the image data, such as voxel size (from metadata), PSF size, noise SNR, as well as the size of the features of interest (eg the size of the visualized cells). The feature size and noise SNR can be determined in Fiji by measuring the size, intensity mean, and SD for a few selected regions. To estimate the size of the PSF, the user will need a PSF image that is either generated on theoretical grounds or measured from imaging fluorescent beads (Figure 5B). In DeconvTest, we provide a tool to compute the PSF SD by approximating it with a 3D Gaussian function.

Once all image properties are determined (Figure 5C), the user can adjust the configuration file (Figure 5D) to generate synthetic microscopy data with the same properties and deconvolve them with specified algorithms and settings. When running the workflow with this configuration file, DeconvTest will generate a csv table with NRMSE values for all tested deconvolution settings (Figure 5E). The user can choose the settings with the lowest NRMSE directly from the table or plot the results to further examine how the deconvolution parameters influence NRMSE. Once the optimal deconvolution parameters are determined (Figure 5F), the users can deconvolve their microscopy images with these parameters from within DeconvTest or using external software (Figure 5G).

4 | CONCLUSION

We presented the open-source Python-based simulation framework DeconvTest, which provides the means to generate synthetic microscopy data, deconvolve the images with different algorithms, and quantify the algorithm performance in a systematic and high-throughput way. The results presented in this work demonstrate the power of the DeconvTest framework and provide a guide for choosing optimal values of deconvolution parameters for two-photon microscopy images. Future framework extensions will enable similar studies for other microscopy techniques, which—we envisage—should result in creating a comprehensive guideline on the choice of deconvolution methods and their parameters for image data with specific properties. Furthermore, DeconvTest paves the way for exploiting the full potential of quantitative microscopy in the future: The comparison of deconvolution results for varying imaging conditions in an automatized fashion will allow microscopy users to achieve optimal results for the combined process of imaging and deconvolution.

Orcid

Marc Thilo Figge 🐱 https://orcid.org/0000-0002-4044-9166

References

[1] P. Sarder, A. Nehorai, Signal Process. Mag. IEEE 2006, 23 (3), 32.
[2] A. Griffa, N. Garin, D. Sage, GIT Imaging Microsc. 2010, 12 (1), 43.
[3] A. Griffa, N. Garin, D. Sage, GIT Imaging Microsc. 2010, 12 (3), 41.
[4] D. Svoboda, M. Kozubek, S. Stejskal, Cytometry A 2009, 75A (6), 494.
[5] D. Wiesner, D. Svoboda, M. Maška, M. Kozubek, Bioinformatics 2019, 35(21), 4531.
[6] V. Ulman et al., Nat. Methods 2017, 14(12), 1141.
[7] H. Kirshner, F. Aguet, D. Sage, M. Unser, J. Microsc. 2013, 249 (1), 13.
[8] R. Dougherty, “Extensions of DAMAS and benefits and limitations of deconvolution in beamforming.” 2005.
[9] D. Sage et al., Methods 2017, 115, 28.
[10] C.-Y. Dong, K. Koenig, P. So, J. Biomed. Opt. 2003, 8(3), 450.
[11] J. Schindelin et al., Nat. Methods 2012, 9(7), 676.
[12] J. Dudeck et al., J Exp Med 2017, 214(12), 3791.

SUPPORTING INFORMATION
Additional supporting information may be found online in the Supporting Information section at the end of this article.