Contents lists available at ScienceDirect

Computational and Structural Biotechnology Journal

journal homepage: www.elsevier.com/locate/csbj

Software/Web server Article

PENGUIN: A rapid and efficient image preprocessing tool for multiplexed spatial proteomics

A.M. Sequeira [a,b], M.E. Ijsselsteijn [b], M. Rocha [a], Noel F.C.C. de Miranda [b,*]

[a] Department of Informatics, School of Engineering, University of Minho, Braga, Portugal
[b] Department of Pathology, Leiden University Medical Centre, Leiden, the Netherlands

ARTICLE INFO

Keywords:
Multiplex imaging
Immunophenotyping
Denoising
Normalization
Background subtraction
Spatial omics

ABSTRACT

Multiplex spatial proteomic methodologies can provide a unique perspective on the molecular and cellular composition of complex biological systems. Several challenges are associated to the analysis of imaging data, specifically in regard to the normalization of signal-to-noise ratios across images and subtracting background noise. However, there is a lack of user-friendly solutions for denoising multiplex imaging data that can be applied to large datasets. We have developed PENGUIN –Percentile Normalization GUI Image deNoising: a straightforward image preprocessing tool for multiplexed spatial proteomics data. Compared to existing approaches, PENGUIN distinguishes itself by eliminating the need for manual annotation or machine learning models. It effectively preserves signal intensity differences while reducing noise, improving downstream tasks such as cell segmentation and phenotyping. PENGUIN's simplicity, speed, and intuitive interface, available as both a script and a Jupyter notebook, make it easy to adjust image processing parameters, providing a user-friendly experience. We further demonstrate the effectiveness of PENGUIN by comparing it to conventional image processing techniques and solutions tailored for multiplex imaging data.

1. Introduction

In recent years, multiplexed imaging technologies have advanced significantly, enabling spatially resolved profiling of biological samples [1,2]. Imaging Mass Cytometry (IMC) [3] and Multiplex Ion Beam Imaging (MIBI) [4] make use of metal-conjugated antibodies for the detection of proteins by means of mass spectrometry. This process involves quantifying isotopic reporter masses released from tissue after the ablation of small regions using a laser beam or ion beams. Other antibody-based multiplex approaches, such as Co-Detection by IndEXing (CODEX) [5], rely on fluorescent-labeled antibodies.

Despite the growing use of multidimensional proteomics spatial technologies, the resulting data presents challenges that conventional image analysis methods often struggle to address. A common challenge is the presence of noise in data, which must be removed before analysis [6]. Noise sources can vary based on the antibodies, detection channels, and tissue types used, and may manifest as artifacts such as hot pixels or background noise [6–8]. Additionally, some immunodetections may exhibit weak signals and low signal-to-noise ratios [9]. Together, these noise factors degrade image quality and complicate downstream analyses of multiplex imaging data. Consequently, robust and reliable denoising methods have become increasingly important as imaging technologies are more widely applied [6].

Specifically, IMC noise includes hot pixels, shot noise and channel crosstalk [10]. Channel crosstalk, where signals from one channel interfere with adjacent channels, can lead to signal misidentification [6, 9]. To address this, methods like CATALYST [8] use pre-acquisition signal compensation matrices, while post-acquisition solutions are also available [7,11]. However, well-designed antibody panels can often reduce the need for correction by minimizing overlap between channels [9,10]. Hot pixels are characterized by individual pixels with significantly higher signal intensities than their surroundings. Additionally, small clusters of consecutive hot pixels, which do not represent biological structures, may form due to nonspecific antibody binding, antibody aggregates, or contamination by dust particles [7,10,12]. Shot noise, resulting from ion counting imaging processes, is pixel-independent but signal-dependent manifesting as random signal variations that typically follow a Poisson distribution [10]. This means noise is more pronounced at lower signal levels.

Hot pixels are commonly addressed using traditional image filtering

* Corresponding author.
E-mail address: n.f.de_miranda@lumc.nl (N.F.C.C. de Miranda).

https://doi.org/10.1016/j.csbj.2024.10.048
Received 25 July 2024; Received in revised form 25 October 2024; Accepted 27 October 2024
Available online 31 October 2024

methods and thresholds. Researchers have also developed numerous "homebrew" computational strategies, customized for specific projects [11,13,14]. One notable approach, implemented in Steinbock, a Python toolkit for processing multidimensional images, mitigates hot pixels by comparing each pixel's value to those of its eight surrounding neighbors. If the difference between the pixel and any of its neighbors exceeds a predetermined threshold, the pixel's value is adjusted to match the highest neighboring pixel value [15].

In addition to conventional denoising techniques, tools like Ilastik provide supervised pixel classification, allowing for the distinction between background noise and true signal pixels for each marker [15–17]. In this approach, an experienced user trains the Ilastik Random Forest pixel classifier by manually labeling pixels as signal or background. Once trained, the model is applied across all images, generating a binary expression map where non-noise pixels are assigned a value of 1. This method has proven effective not only in removing background noise but also in standardizing and normalizing signals across samples, thereby reducing batch effects [17]. However, the requirement for manual annotation across images can be labor-intensive. This manual curation process is particularly challenging when dealing with large datasets, making this approach less practical for large datasets.

Recently, Lu et al. [10] introduced IMC-Denoise, a two-step pipeline that combines a Differential Intensity Map-based Restoration (DIMR) algorithm to remove hot pixels and a self-supervised deep learning (DL) algorithm called DeepSNiF for filtering shot noise. DeepSNiF, inspired by the widely used Noise2Void algorithm, does not rely on pretrained data. Instead, it trains a neural network to eliminate noise by leveraging information from surrounding pixels within the same image, eliminating the need for a separate, clean reference image. The process begins by identifying hot pixels using Anscombe transformation differential maps. Then, DeepSNiF filters shot noise by randomly masking pixels using a stratified sampling approach. However, DL approaches like this are often time-consuming, resource-intensive and require careful parameter tuning.

Despite the availability of various IMC denoising methods, there is still no consensus on the optimal approach, and the lack of a standardized strategy hinders comparisons between datasets. Current methods either require the manual identification of noisy pixels across all markers – a time-consuming and inconsistent process – or fail to adequately address all types of noise and nonspecific antibody signals.

In this work, we introduce PENGUIN (Percentile Normalization GUI Image denoising), an enhanced pipeline for denoising multiplexed proteomics data. PENGUIN employs scaling, thresholding, and percentile-based filters to address various noise sources in multiplex images, including hot pixels and shot noise. The method is fast, scalable, and reproducible, without requiring manual pixel annotation or extensive hardware resources. We demonstrate its effectiveness using an existing IMC dataset, benchmarking it against current preprocessing methods and those specifically designed for multiplex imaging data analysis. Additionally, we show its versatility by applying PENGUIN to multiplex immunofluorescence (IF) images, illustrating its applicability across different imaging modalities. The tool is available on https://github.com/deMirandaLab/PENGUIN.

2. Results

To mitigate noise in IMC images, the following considerations were identified:

1. Hot pixels, characterized by random pixels exhibiting significantly higher signal intensities than their surroundings, can be efficiently addressed by filtering out sparse signals, specifically pixels that lack neighboring signals (Supplementary Figure 1).
2. Shot noise, arising from inherent variability in photon detection, can result in low-intensity signals in regions where no specific signal is expected. This noise is particularly prominent in areas with weak or no biological signal and can produce non-zero pixel values due to random detection fluctuations (Supplementary Figure 1).
3. Noise typically occurs independently across different channels within the same image, meaning one channel can be affected without others being impacted. However, within a specific cohort, noise generally follows consistent patterns across all images from the same channel.

Based on these considerations, PENGUIN was developed to integrate scaling, thresholding, and percentile-based filters for denoising IMC data (Fig. 1 A). PENGUIN follows a multistep approach: 1) Images are saturated at the 99th percentile to reduce the impact of extremely bright pixels; 2) Signal intensities across each channel and image are normalized to a scale from 0 to 1; 3) An adaptable threshold is applied to remove low-intensity signals; and 4) Hot pixels are identified using adaptable percentile filters and removed. A key feature of this strategy is that the percentile filter is used solely for hot pixel identification, ensuring their removal without affecting image clarity or edge definition. As a result, post-processed images maintain sharpness and edge integrity. Additionally, the normalization of values from 0 to 1 simplifies their direct use in subsequent analyses, such as cell segmentation.

To streamline the visualization and deployment of the preprocessing pipeline, PENGUIN is provided as a user-friendly Jupyter notebook (Fig. 1B). This tool allows users to directly observe how thresholds and percentiles impact different markers, aiding in the selection of the most suitable thresholds for each marker and image. The code for the preprocessing pipeline is freely available, allowing users to also apply the functions directly via scripting.

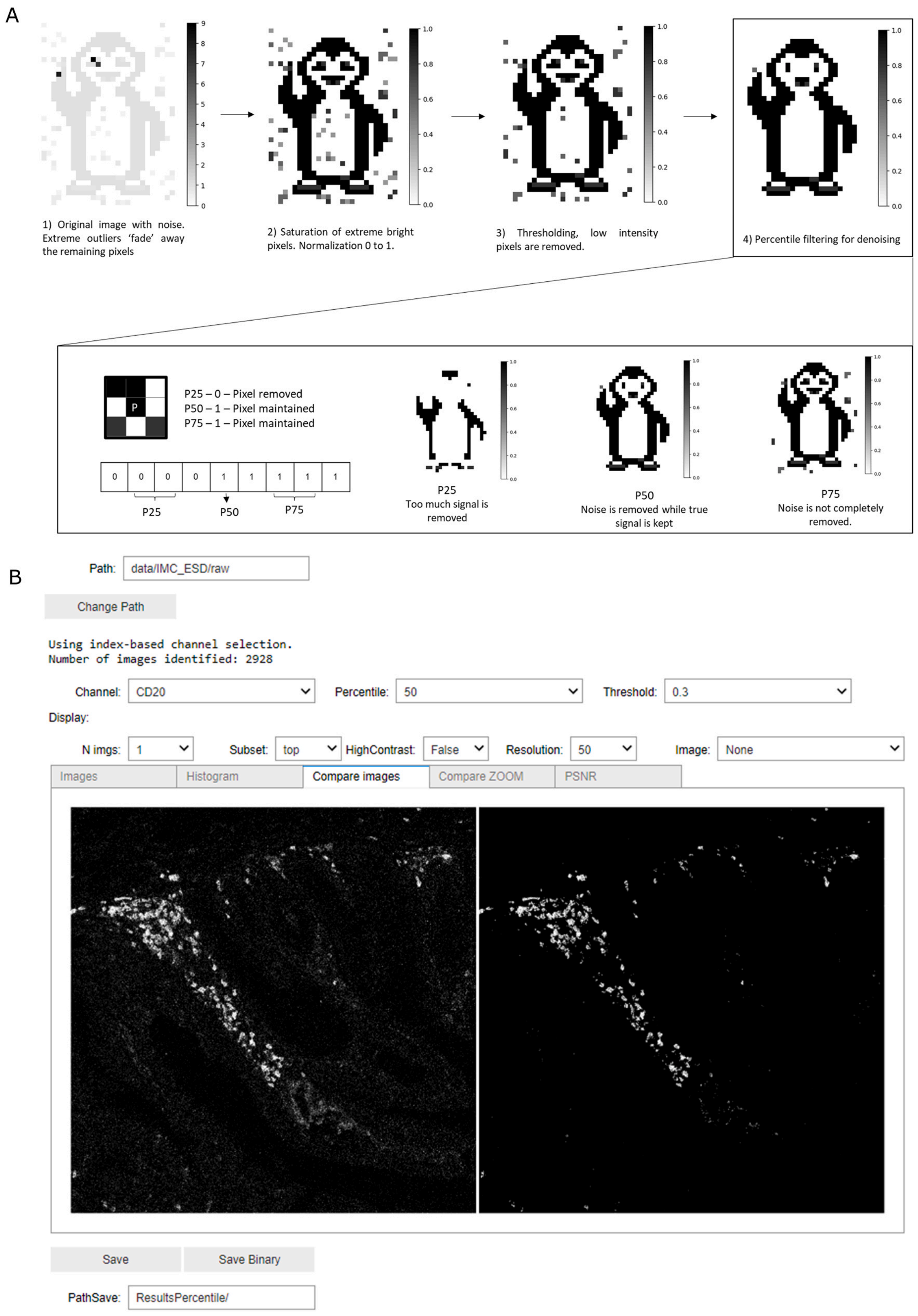
Key adjustable parameters in this pipeline include the Threshold (T) value and the Percentile (P) value, both of which are critical to its performance (Fig. 2). A higher T value removes more signal during the initial phase of the pipeline. If a channel primarily contains background signal, with only high-intensity values representing true signals, T should be set high. Conversely, in channels with minimal background noise, T can be lower or even omitted. The P parameter plays a crucial role, with lower values (e.g., 25) being stricter, removing nearly all sparse signals, while higher values (e.g., 75) are more lenient, allowing more sparse signals to remain. Tailoring these parameters for each channel is essential due to the significant variability in marker behavior. For example, β-catenin and FOXP3 showed optimal noise reduction with a T of 0.1 and a P of 50, while CD20 required a higher threshold of T 0.3 to effectively eliminate noise. In contrast, clear images for vimentin and CD45 were achieved with P set at 25, without needing a threshold setting (Fig. 2).

2.1. Comparison of available methods for IMC denoising

PENGUIN was evaluated against both traditional image denoising techniques and IMC-specific methods using a public dataset of 39 cellular markers analyzed by IMC across 61 samples [18].

To enhance the performance of standard image denoising methods, we applied saturation at the 99th percentile to remove bright outliers and normalized each channel's signal intensity between 0 and 1 [17]. We then tested a variety of classical filtering methods, including Gaussian, mean, percentile, non-local means, bilateral, total variation, wavelet, anisotropic diffusion, and BM3D filters (see Supplementary Figures 1& 2). As expected, linear filters like Gaussian and mean filters tended to blur the images, causing a loss of detail and definition in tissue boundaries. While these filters did not directly eliminate hot pixels, they made the pixels less noticeable by blurring them. The other classical filters either failed to effectively reduce noise (e.g., anisotropic and bilateral filters) or distorted the images, as seen with the non-local mean filter, making them unsuitable for this type of data.

We also evaluated DL methods designed for denoising images without requiring paired ground truth data, specifically Noise2Void [19] and denoising autoencoders [20], trained for 20 or 50 epochs.


A
1) Original image with noise. Extreme outliers 'fade' away the remaining pixels
2) Saturation of extreme bright pixels. Normalization 0 to 1.
3) Thresholding, low intensity pixels are removed.
4) Percentile filtering for denoising
P
P25 – 0 – Pixel removed
P50 – 1 – Pixel maintained
P75 – 1 – Pixel maintained
0 0 0 0 1 1 1 1 1
P25
P50
P75
P25
Too much signal is removed
P50
Noise is removed while true signal is kept
P75
Noise is not completely removed.
B
Change Path
Images
Histogram
Compare images
Compare ZOOM
PSNR
Save
Save Binary

Path: data/IMC_ESD/raw

```
Using index-based channel selection.
Number of images identified: 2928
```

Channel: CD20 Percentile: 50 Threshold: 0.3

Display:

N imgs: 1 Subset: top HighContrast: False Resolution: 50 Image: None

PathSave: ResultsPercentile/

Fig. 1. - Overview of the PENGUIN tool. (A) Overview of the PENGUIN pipeline: 1 - the original image contains noise and extreme outliers, which obscure the rest of the signal. 2 - outliers are managed by capping values above the 99th percentile, enhancing visualization. Each channel and image are then normalized to a scale of 0 to 1. 3 - pixels with values below a selected threshold are removed. 4 - percentile filtering is applied to detect and remove noise pixels. For each central pixel, a 3 × 3 window is considered and the pixel values are sorted in ascending order. If the pixel is at the 0th percentile, it is classified as noise and set to 0; if it is above 0, it is considered part of the signal. For example, a 50th percentile (P50) requires that 5 out of 9 pixels be positive, the 75th percentile requires 2 out of 9, and the 25th percentile (P25) requires 7 out of 9. Depending on the characteristics of the channel, low percentile values may result in the removal of true signals, while high percentile values may retain noise. (B) Sneak peek of the PENGUIN Jupyter notebook.

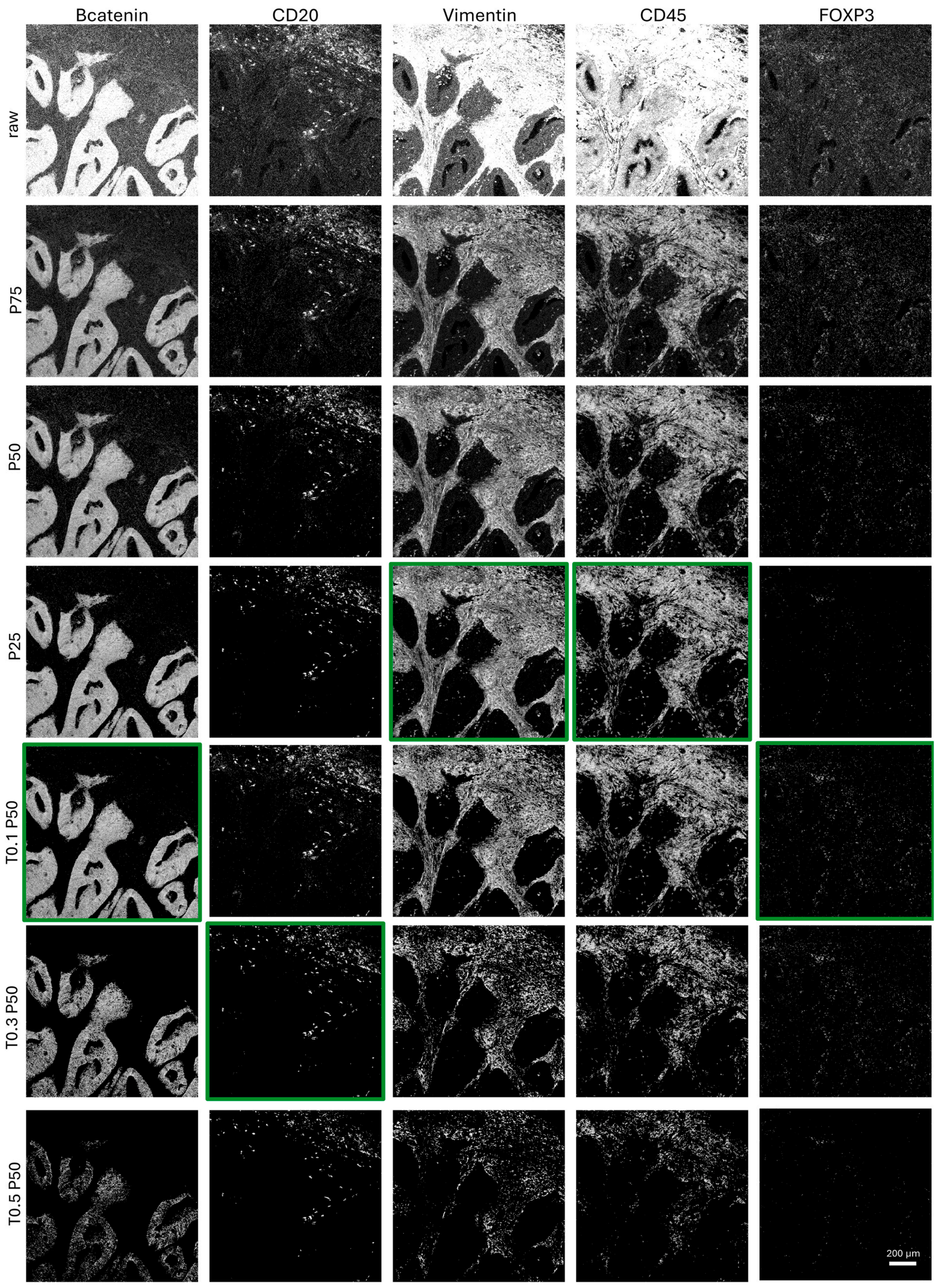


Fig. 2. - Application of various percentiles (P90, P75, P50, P25, P10) and thresholds (T0.1, T0.3, T0.5 with P50) with PENGUIN were used to assess immunodetection of b-catenin, CD20, vimentin, CD45, and FOXP3. The optimal parameters for each marker are highlighted in green.

These methods did not effectively denoise the IMC images and led to image distortion, suggesting they may not be well-suited for this type of data. However, it is important to note that DL models tend to perform better with larger datasets, and as more IMC image datasets become available, the performance of these models may improve. Additionally, our experiments only employed a basic convolutional denoising autoencoder, leaving room for significant optimization through different combinations of layers, epochs, and parameters. However, DL methods are time-intensive, requiring considerate effort for development, tuning, and training, and often necessitate specialized hardware, which that may not be readily accessible to all users.

Next, we applied two IMC-specific methodologies. First, we tested a hot pixel filter [15] that uses a 3 $\times$ 3 kernel to determine if the difference between a pixel and its maximum neighboring value exceeds a predefined threshold; if so, the pixel is classified as noise and replaced with the maximum neighbor's value. This method was applied to both normalized images, using thresholds ranging from 0.05 to 0.2, and raw images with a default threshold of 50. While this approach successfully removed some hot pixels, the processed images still contained noise, and in some cases, real signals were lost (Supplementary Figures 1& 2). Second, we applied IMC Denoise [10], using both the DIMR algorithm alone and in combination with the DeepSNIF DL algorithm, following the authors' guidelines and default settings. The DIMR algorithm alone proved insufficient for fully removing hot pixels. When combined with DeepSNIF, similar issues arose as with Noise2Void, including image distortions. As with other DL methods, the possible configurations for IMC Denoise are virtually limitless; however, fine-tuning the optimal parameters is a labor-intensive process. Additionally, separate models must be trained for each channel, further complicating the implementation of this approach.

To evaluate IMC denoising performance with PENGUIN, we employed our lab's semi-automated background removal (SABR) pipeline [17], which utilizes Ilastik (Fig. 3 & Supplementary Figure 4). Given the lack of a true ground truth, we used the manually annotated SABR-denoised dataset as a reference standard. Using this dataset, we calculated the Peak Signal-to-Noise Ratio (PSNR), where higher values denote reduced error, and the Structural Similarity Index (SSIM), which ranges from -1 to 1, with a score of 1 indicating perfect similarity (Supplementary Figure 5). These metrics enabled a comparative analysis of the denoising performance of raw (rescaled) and PENGUIN-denoised images relative to the SABR dataset, our designated reference. Results demonstrated a marked improvement in image quality over raw data, with PENGUIN's denoising closely aligning with the SABR-annotated dataset. Furthermore, the signal distributions across raw, 99th percentile normalized values, SABR, and PENGUIN-denoised images are presented in Supplementary Figures 6, 7, and 8. It is worth noting that SABR denoising may occasionally result in signal loss for certain markers, as observed with β-catenin immunodetection (Fig. 3 and Supplementary Figure 4). By applying PENGUIN to the same image set, we observed effective noise reduction akin to that achieved by the SABR pipeline, with the added advantage of retaining more signal than SABR.

In summary, PENGUIN effectively eliminates noise from IMC images. While it requires manual adjustment of thresholds and percentiles for each channel, it does not necessitate developing specific classifiers to differentiate between signal and noise for each marker. Additionally, the settings are marker-dependent and often translatable between images, making it easy to apply to new images obtained with the same antibody panel. Unlike the Ilastik-based method, which produces binary images indicating only the presence or absence of signals, PENGUIN generates images with normalized values ranging from 0 to 1, capturing a range of signal intensities. This allows for more nuanced analysis, as PENGUIN not only retains more signal but also preserves critical details of cellular structures.

Setting the thresholds and percentiles is straightforward, further

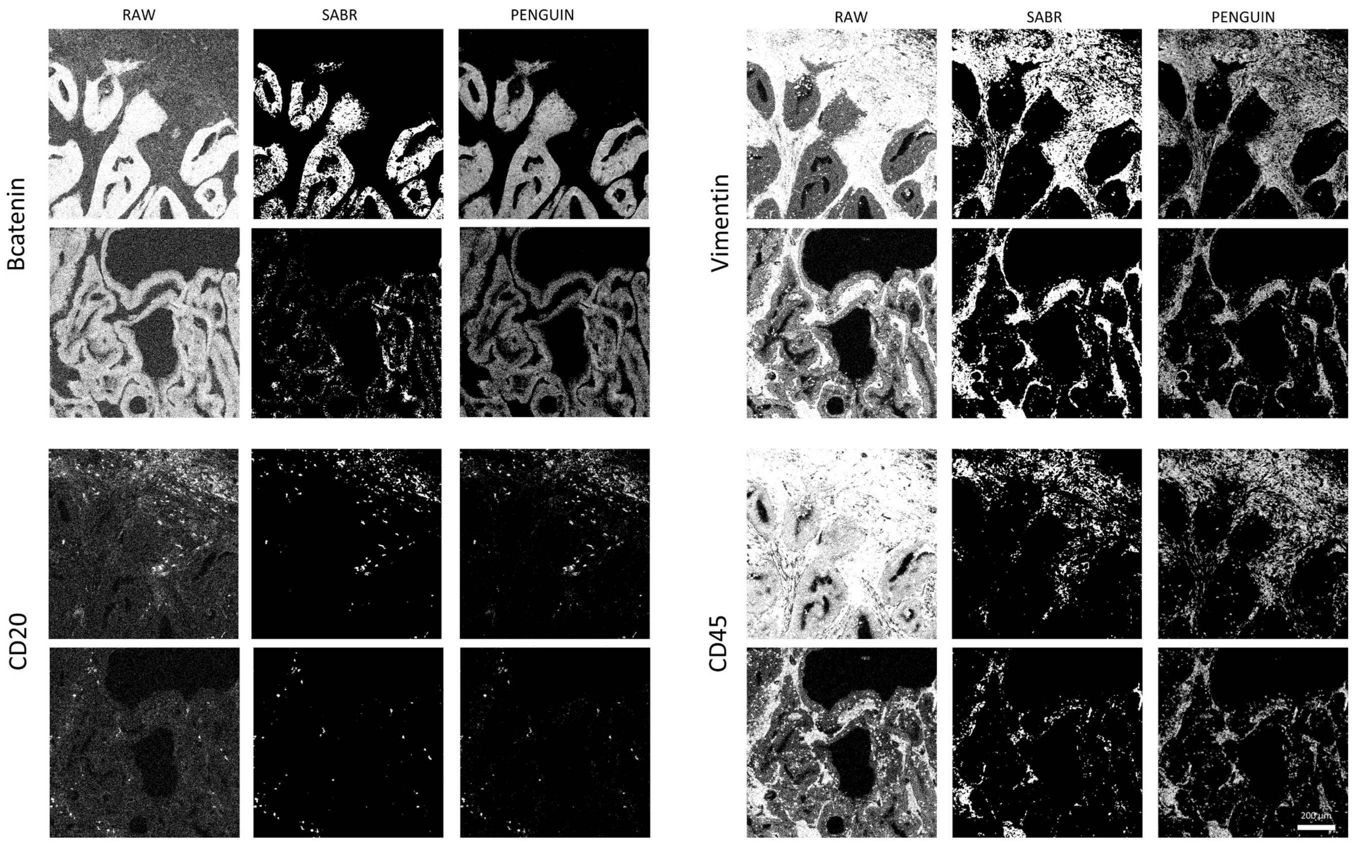


Fig. 3. - β-catenin, Vimentin, CD20, and CD45 immunodetection in two regions of interest, showing raw images, processed by the Ilastik-based SABR approach, and PENGUIN.

simplified by the provided notebook. Processing all 61 images per channel took under 4 s on an Intel(R) Core(TM) i7–9700 CPU – a major improvement over Ilastik, which requires 5 to 15 min per marker. PENGUIN processing itself takes about 4 s per channel, with an additional 5 min for expert parameter selection. In contrast, SABR requires approximately 10 min of algorithm training plus 5 min per channel for processing. This highlights PENGUIN's efficiency over both Ilastik and SABR. Additionally, without the need for model training, PENGUIN's results are fully reproducible and readily accessible, offering a significant improvement over existing methods.

2.2. PENGUIN enhances downstream analysis

After confirming the effectiveness of PENGUIN for image processing, we evaluated its impact on downstream analysis. This included comparing raw, unprocessed images with those corrected using the Ilastik-based SABR method and the PENGUIN pipeline, focusing on cell segmentation and cell phenotyping/clustering.

A key advantage of PENGUIN over previous approaches is that its output can be directly used in cell segmentation pipelines, eliminating the need to train an additional Ilastik model to distinguish between membrane, nucleus, and background markers, thereby saving significant time. Furthermore, as highlighted earlier and shown in Fig. 4, PENGUIN's normalization process preserves more signal and retains important cellular features.

To compare these methods, we combined the normalized images with their corresponding cell segmentation masks to calculate relative marker expression per cell. Cells were then clustered using t-SNE to identify cell subsets (Fig. 4A). Consistent with findings by Ijsselsteijn et al. [17], normalization and background removal are crucial for accurately identifying signals and cells, as well as normalizing inter-sample variation in IMC data for automated downstream analysis. Like current state-of-the-art techniques, PENGUIN adjusts for inter-patient variability, resulting in well-defined cells and clusters. Next, both the SABR and PENGUIN-denoised datasets were phenotyped (Fig. 4B). PENGUIN-normalized cellular phenotypes were easily identifiable and displayed clear marker patterns, comparable to those in SABR-normalized data. Additionally, mapping these phenotypes back onto the images demonstrated that cellular distribution was consistent with the original data (Fig. 4C).

2.3. PENGUIN can be applied to other image modalities

Various imaging modalities in multiplex spatial proteomics, such as MIBI, CODEX, and multiplex IF, produce different types of images. Encouraged by the successful outcomes of the PENGUIN pipeline, and given that these modalities share similar limitations, we extended the application of PENGUIN to a set of in-house generated IF images (Fig. 5). The results demonstrated high-quality denoised images, achieved with significantly reduced time, resources, and operator-related variability.

3. Discussion

Spatial proteomics data, such as that generated by IMC, offers unique insights into the spatial organization of tissues, with broad applications. However, managing signal-to-noise ratios, especially with large antibody panels, presents significant challenges. Therefore, effective computational preprocessing methods are crucial to ensuring the reliability of subsequent analyses [6]. While various custom algorithms have been successful in denoising IMC data, the growing volume of data calls for a standardized and efficient preprocessing pipeline.

In this context, we introduce PENGUIN, a denoising pipeline that effectively processes IMC images using scaling, thresholding, and percentile filters, yielding excellent results. We compared PENGUIN to traditional filtering methods, DL-based approaches, and IMC-specific solutions, demonstrating that it delivers clear, high-quality images quickly, easily, and reproducibly. PENGUIN is available as a user-friendly Jupyter notebook, enabling the simple adjustment of thresholds and percentiles for each marker and streamlining the transformation and saving of images. The core code is also available for direct scripting.

Compared to the recently developed Ilastik-based SABR method, PENGUIN eliminates the need for manual annotation or machine learning model training, preserves signal intensity variations across tissues, reduces noise, and retains more structural details. Additionally, PENGUIN improves critical downstream tasks like cell segmentation and phenotyping. Due to its simplicity, speed, intuitive interface, and reproducibility, PENGUIN is a valuable tool for the analysis of IMC data and, potentially, other multiplex imaging platforms.

4. Methods

All the tests were run in a Intel(R) Core(TM) i7–9700 CPU. The DL models were run using a GPU Nvidia TU102 GeForce RTX 2080 Ti.

4.1. Dataset

All the results for IMC were obtained using the publicly available cohort of 61 IMC raw images with 39 markers of colorectal cancer [18].

4.2. Denoising Methods comparison

Before application of the denoising filters, images were saturated at the 99th percentile and normalized between 0 and 1. Raw images were also tested, but normalization led to significant improvements in image quality.

The following filters were applied to each channel: Gaussian filter with sigma values of 0.7 and 1; mean filter; percentile filter with values of 25, 50, and 75; non-local means filter using a patch size of 5, patch distance of 11, and sigma of 0.2); bilateral filter; total variation filter with a weight of 0.3; Wavelet filter; anisotropic diffusion filter and BM3D filter with sigma values of 0.1, 0.2 and estimated sigma values between these two. For parameters not explicitly specified, default settings were used. Gaussian, mean, and percentile filters were implemented using SciPy [21], while non-local means, bilateral, total variation and wavelet filters were based on the scikit-image package [22]. BM3D filtering was retrieved from its source package [23].

The hot pixel removal filter for IMC images was implemented following Windager et al. [15], using a Scipy maximum filter with a 3×3 window. If the difference between the central pixel and its maximum neighbor exceeded a defined threshold, the central pixel was set to the value of the maximum neighbor. Thresholds ranging from 0.05 to 0.2 were applied to normalized images, while the default threshold of 50 was used for raw images.

Noise2Void was trained per channel as described by Krull et al. [19], with 0.2 of the data used for validation. Loss was defined as MSE, with a neighborhood radius 5 and UNET kernel size 3 for a patch size of 64. Models were trained for 20, 50, and 100 epochs.

The convolutional autoencoder for image denoising were built as described by Chollet et al. [20]. The encoder consisted of two consecutive 2D convolutional layers (32 filters, kernel size 3×3, ReLU activation) followed by max pooling (pool size 2×2). The decoder included two transpose convolutional layers (32 filters, kernel size 3×3, ReLU activation) and a final convolutional layer with 1 filter and sigmoid activation. The Adam optimizer and binary cross-entropy loss function were used. Training was performed for 20 or 50 epochs per channel. The DL implementation was done using Tensorflow [24], with any unspecified parameters set to default.

IMC Denoise [10] was implemented according to the instructions in the tutorial notebook, with default parameters. IMC Denoise consists of two steps: the DIMR algorithm and DeepSNIF. We tested both the DIMR algorithm applied per channel (using 4 neighbors, a window size of 3,

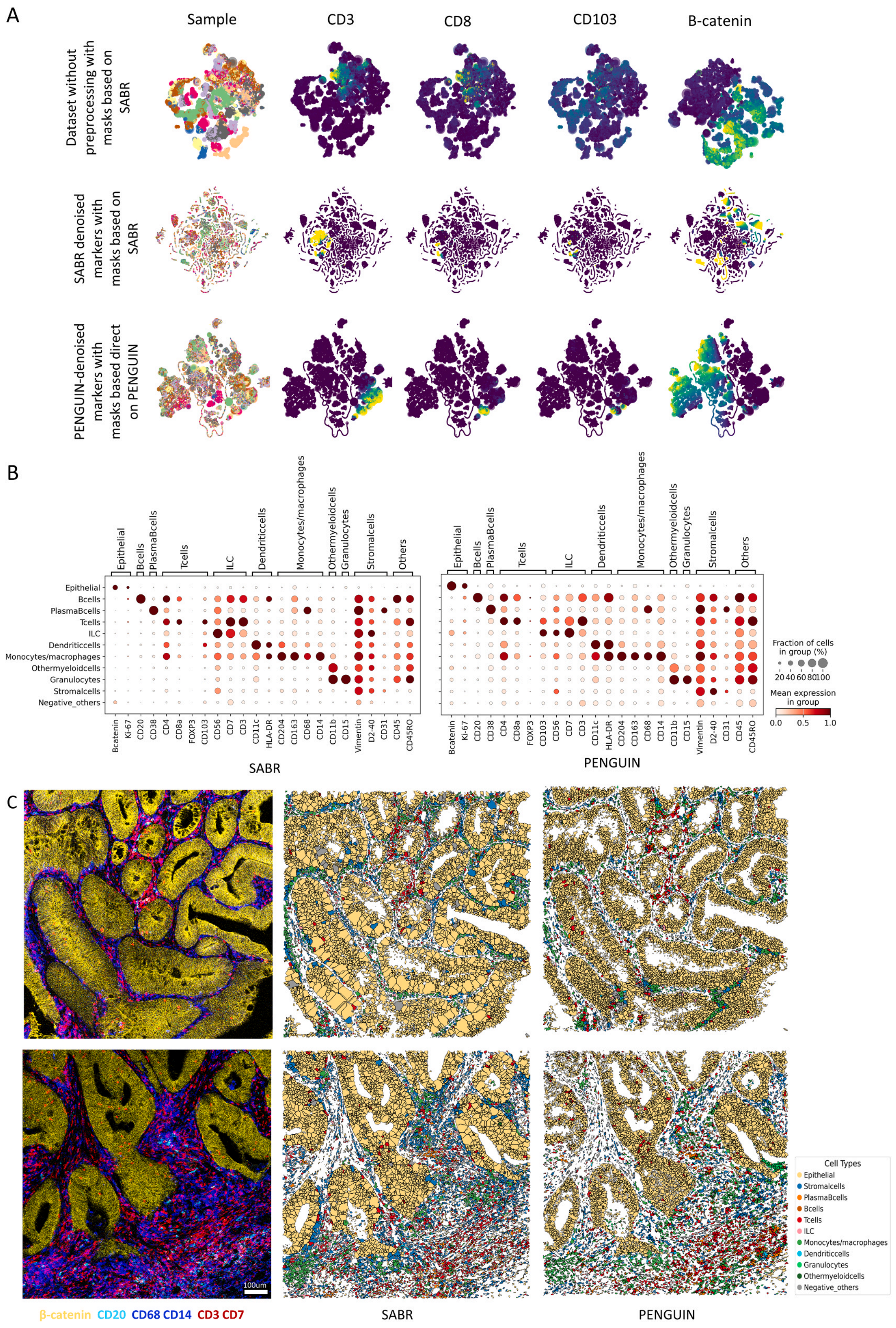


Fig. 4. - Overview of downstream analysis. A) t-SNE plots generated from raw marker expression with SABR Ilastik-based cell masks(top), SABR Ilastik-denoised marker expression with SABR Ilastik-based cell masks (middle), and PENGUIN-denoised marker expression and PENGUIN-based cell masks (bottom). The t-SNE plots are colored by sample, CD3, CD8, CD103 and b-catenin expression levels. B) Dot plot showing the scaled expression of marker genes across epithelial, stromal, immune lymphoid lineages (B cells, plasma B cells, T cells, innate lymphoid cells (ILC)) and immune myeloid lineages (dendritic cells, monocytes, macrophages, and granulocytes) for each cell type identified using cell segmentation and marker expression from SABR and PENGUIN denoising strategies. C) Comparison of cell segmentation and phenotype obtained using SABR and PENGUIN denoising strategies within a sample.

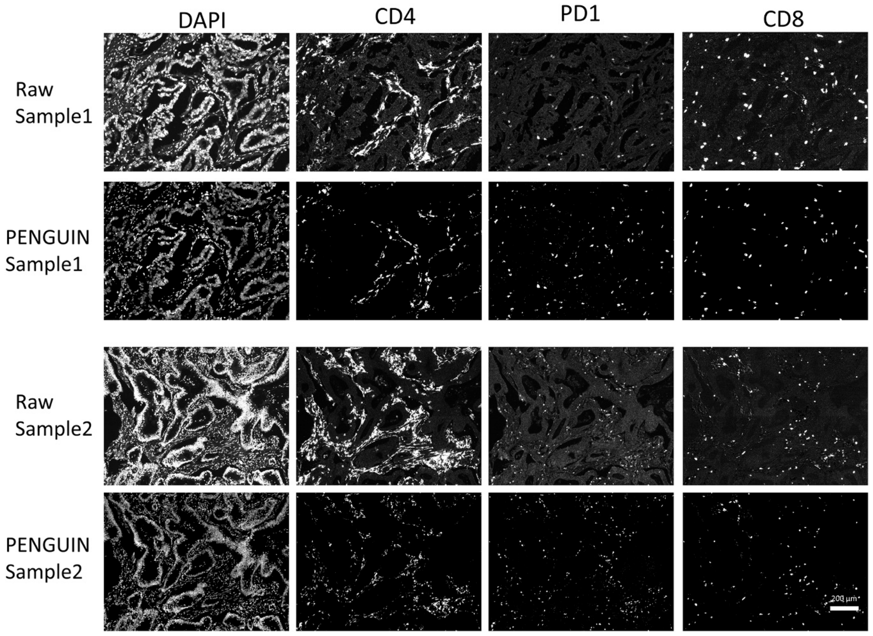


Fig. 5. - PENGUIN applied to IF images, specifically for DAPI, CD4, PD-1 and CD8 immunofluorescent detection.

and 3 iterations) and the combination of the DIMR algorithm followed by the DeepSNIF, which was trained for 50 epochs.

4.3. PENGUIN implementation

PENGUIN is implemented in Python 3.10 and available as both a code package and Jupyter notebook. For each image and channel, pixels above the 99th percentile are capped at that value and normalized to a scale of 0 to 1. Following this, pixels with values below a defined threshold are removed, and a percentile filter (using a 3 ×3 window) is applied to determine which pixels are either considered noise or signal. The same thresholds and percentiles are applied consistently across all images in the cohort for each channel. To demonstrate the approach, we tested percentile values of 90, 75, 50, 25, and 10, as well as the 50th percentile after removing pixels below thresholds of 0.1, 0.3, and 0.5. This comparison was performed for β-catenin, CD20, vimentin, CD45, and FOXP3, given their importance as markers and varying signal distributions. Two Jupyter notebooks are available: one for cases where each region of interest has a separate TIFF file per channel, and another for cases where all channels are contained in a single TIFF stack.

4.4. Comparison of methods for downstream analysis

4.4.1. PENGUIN parameters definition

The raw images were analyzed by an IMC expert using PENGUIN notebook for better visualization. Parameters such as thresholds and percentiles were adjusted for each channel using the Notebook tool (Supplementary Table 1) and then applied uniformly across all images in the cohort.

4.4.2. Ilastik-based background removal

For each marker, a random forest classifier was trained in Ilastik [16] to distinguish between background and signal pixels, as described by Ijsselsteijn et al. [17]. After thorough training on approximately 10 % of the dataset, the classifier was applied to all images for a given marker. The data was then exported as simple segmentation masks, with background pixels set to zero and signal pixels set to one.

4.4.3. Denoising evaluation metrics

To evaluate the quality of denoising in the absence of an actual ground truth, we used SABR [25] as a reference, given its manual annotation and reliability as a close approximation. We calculated the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) to compare the performance of rescaled raw and PENGUIN-denoised images against the SABR output, treated as the reference standard. In order to make this comparison, positive pixels in SABR (i.e., pixels considered signal) were replaced with their original values from the rescaled raw image. PSNR is defined as:

$$PSNR = 10 \quad \bullet \log 10 \quad \left(\frac{MAX_I^2}{MSE} \right)$$

where MAX_I is the maximum possible pixel value of the image, and MSE is the mean squared error.

SSIM is defined as:

$$SSIM(x, y) = \quad (l(x, y))^{\alpha} \bullet (c(x, y))^{B} \bullet (s(x, y))^{\gamma}$$

where luminance (l) is measured by averaging pixel values, contrast (c) is calculated as the standard deviation of pixel values, and structure (s) is derived from the covariance between the signals, normalized by the standard deviation. The parameters $\alpha > 0$, $\beta > 0$, $\gamma > 0$ represent the relative importance of each component. We used the scikit-image package for both PSNR and SSIM calculations.

4.4.4. Cell segmentation & phenotyping

Cell masks were created using CellProfiler V3.0 [26]. For comparison, masks were generated from both raw data and PENGUIN-normalized images. For raw data, a combination of Ilastik/CellProfiler was applied: first, probability masks for the nucleus, membrane, and background were generated in Ilastik using the DNA, vimentin and keratin images. These masks were then loaded into a CellProfiler pipeline to identify primary objects (nuclei), followed by membrane assignment using the “Identify Secondary Objects” module. Visual inspection was conducted to compare the masks with the original IMC Images. After PENGUIN normalization, the normalized images could be directly used to create cell segmentation masks in Ilastik using the same workflow.

The two different cell segmentation masks (Ilastik-based and PENGUIN-based) were loaded into ImaCytE [27], along with the raw, Ilastik-denoised, and PENGUIN-denoised marker profiles, to define relative marker expression per cell. FCS files were generated, and clustering of cells was performed using t-SNE in Cytosplore [28] to identify cell subsets.

4.5. Immunofluorescence labelling and imaging

Immunofluorescence labelling was performed using the OPAL methodology. Briefly, four µm thick sections from formalin-fixed paraffin-embedded tissues underwent deparaffinisation, rehydration, endogenous peroxidase blocking, and antigen retrieval with 10 mM citrate buffer (pH 6.0). Sections were then blocked with Superblock solution (Thermo Fisher Scientific) and incubated for 1 h with anti-CD4. Opal amplification was performed by consecutive 10-minute incubations with a polymeric HRP-linker antibody conjugate (Immunologic, the Netherlands) and OPAL650 reagent (Akoya Biosciences). The sections were boiled for 15 min in 10 mM citrate buffer (pH 6.0). This process of blocking, primary antibody incubation, followed by HRP and OPAL amplification, was repeated for each target (Supplementary Table 2). Between each step, sections were washed with PBS-Tween. Finally, the sections were incubated with 1 µM DAPI, washed with PBS (without Tween) and mounted using Prolong® Gold Antifade Reagent (Cell Signaling Technologies). The slides were imaged and spectrally unmixed using the Vectra 3 imaging system and the InForm software (Akoya Biosciences), after which the component images were normalised with PENGUIN. Parameters, including thresholds and percentiles, were set for each channel (Supplementary Table 2) and applied across all images in the cohort.

Author statement

All authors approve of this manuscript and declare that the manuscript is not under consideration elsewhere.

CRediT authorship contribution statement

Miguel Rocha: Writing – review & editing, Supervision, Funding acquisition. **Noel F.C.C. de Miranda:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Ana Marta Sequeira:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Formal analysis. **Marieke E. Ijsselsteijn:** Writing – review & editing, Validation, Formal analysis.

Declaration of Competing Interest

The authors declare no conflicts of interests.

Acknowledgements

A.M.S. was supported by the Portuguese Foundation for Science and Technology-FCT and FSE through Ph.D. fellowship:2020.0786BD. N.F. C.C.d.M. is funded by the European Research Council under the European Union's Horizon 2020 Research and Innovation Programme (grant agreement no. 852832)) and by VIDI ZonMW (project number 09150172110092).

Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.csbj.2024.10.048.

Data availability

The IMC data used in this study is publicly available from Roelands et al. [18] and can be accessed through the BioImage Archive (Accession number S-BIAD587, https://www.ebi.ac.uk/biostudies/bioimages/studies/S-BIAD587). The PENGUIN pipeline is fully available and ready to use at https://github.com/deMirandaLab/PENGUIN. The repository also contains all the necessary code to replicate the methods comparison, along with detailed Jupyter notebooks for easy understanding and implementation

References

[1] Bressan D, Battistoni G, Hannon GJ. The dawn of spatial omics. Science 2023;381 (6657). eabq4964.
[2] de Vries NL, et al. Unraveling the complexity of the cancer microenvironment with multidimensional genomic and cytometric technologies. Front Oncol 2020;10: 1254.
[3] Giesen C, et al. Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. Nat Methods 2014;11(4):417. 22.
[4] Keren L, et al. MIBI-TOF: a multiplexed imaging platform relates cellular phenotypes and tissue structure. Sci Adv 2019;5(10):eaax5851.
[5] Goltsev Y, et al. Deep profiling of mouse splenic architecture with CODEX multiplexed imaging. Cell 2018;174(4):968–81. e15.
[6] Baharlou H, et al. Mass cytometry imaging for the study of human diseases-applications and data analysis strategies. Front Immunol 2019;10:2657.
[7] Baranski A, et al. MAUI (MBI Analysis User Interface)-an image processing pipeline for multiplexed mass based imaging. PLoS Comput Biol 2021;17(4):e1008887.
[8] Chevrier S, et al. Compensation of signal spillover in suspension and imaging mass cytometry. Cell Syst 2018;6(5):612–20. e5.
[9] Milosevic V. Different approaches to imaging mass cytometry data analysis. Bioinforma Adv 2023;3(1):vbad046.
[10] Lu P, et al. IMC-Denoise: a content aware denoising pipeline to enhance imaging mass cytometry. Nat Commun 2023;14(1):1601.
[11] Wang YJ, et al. Multiplexed in situ imaging mass cytometry analysis of the human endocrine pancreas and immune system in type 1 diabetes. Cell Metab 2019;29(3): 769–83. e4.
[12] Milosevic V. Different approaches to imaging mass cytometry data analysis. Bioinform Adv 2023;3(1):vbad046.
[13] Keren L, et al. A structured tumor-immune microenvironment in triple negative breast cancer revealed by multiplexed ion beam imaging. Cell 2018;174(6): 1373–87. e19.
[14] Rendeiro AF, et al. The spatial landscape of lung pathology during COVID-19 progression. Nature 2021;593(7860):564–9.
[15] Windhager J, et al. An end-to-end workflow for multiplexed image processing and analysis. Nat Protoc 2023;18(11):3565–613.
[16] Berg S, et al. ilastik: interactive machine learning for (bio)image analysis. Nat Methods 2019;16(12):1226–32.
[17] Ijsselsteijn ME, et al. Semi-automated background removal limits data loss and normalizes imaging mass cytometry data. Cytom A 2021;99(12):1187–97.
[18] Roelands J, et al. Transcriptomic and immunophenotypic profiling reveals molecular and immunological hallmarks of colorectal cancer tumourigenesis. Gut 2023;72(7):1326–39.
[19] Krull A, Buchholz T-O, Jug F. Noise2void-learning denoising from single noisy images. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition 2019.
[20] Chollet F. Building autoencoders in keras. The Keras Blog 2016:**14**.
[21] Virtanen P, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 2020;17(3):261–72.
[22] van der Walt S, et al. scikit-image: image processing in Python. PeerJ 2014;2:e453.
[23] Dabov K, et al. Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Trans Image Process 2007;16(8):2080. 95.
[24] Abadi, M., et al., Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467; 2016.
[25] Ijsselsteijn ME, et al. Semi-automated background removal limits data loss and normalizes imaging mass cytometry data. Cytom A 2021.
[26] McQuin C, et al. CellProfiler 3.0: next-generation image processing for biology. PLoS Biol 2018;16(7):e2005970.
[27] Somarakis A, et al. ImaCytE: visual exploration of cellular micro-environments for imaging mass cytometry data. IEEE Trans Vis Comput Graph 2021;27(1):98–110.
[28] Höllt T, et al. Cytosplore: interactive immune cell phenotyping for large single-cell datasets. In: Computer Graphics Forum. Wiley Online Library; 2016.