Published in Image Processing On Line on 2024–07–26.
Submitted on 2024–05–27, accepted on 2024–06–22.
ISSN 2105–1232 © 2024 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2024.549

A Brief Evaluation of InSAR Phase Denoising and Coherence Estimation with Φ-Net

Roland Akiki[1,3], Jérémy Anger[1,3], Carlo de Franchis[1,3], Gabriele Facciolo[1], Raphaël Grandin[2], Jean-Michel Morel[4]

[1]Université Paris-Saclay, ENS Paris-Saclay, Centre Borelli, France
{roland.akiki, jeremy.anger, carlo.de-franchis, gabriele.facciolo}@ens-paris-saclay.fr
[2]Institut de Physique du Globe de Paris, Université Paris VII, France
grandin@ipgp.fr
[3]Kayrros SAS
[4]City University of Hong Kong
jeamorel@cityuedu.hk

Communicated by Pablo Musé   *Demo edited by* Roland Akiki

Abstract

In this article, we examine the joint InSAR phase denoising and coherence estimation performance of the network known as Φ-Net [Sica et al., IEEE Transactions on Geoscience and Remote Sensing, 2021]. We briefly examine the method, network architecture, training data and strategy. Then, in the experimental section, we compare the network's performance against the simple boxcar uniform filter. We verify the observations made by the authors, in particular concerning the superior denoising performance and preservation of fine details in the coherence estimation. Our experiments also indicate that an end-to-end deep learning method might bring a small improvement to the patch-based approach adopted in Φ-Net.

Source Code

The source code and documentation for this algorithm are available from the web page of this article[1]. Usage instructions are included in the README.md file of the archive. The authors' original method implementation is available here[2].
This is an MLBriefs article, the source code has not been reviewed!

Keywords: phase denoising; coherence estimation; CNN; InSAR; demo

[1] https://doi.org/10.5201/ipol.2024.549
[2] https://github.com/DLRRadarScienceGroup/Phi-Net

Roland Akiki, Jérémy Anger, Carlo de Franchis, Gabriele Facciolo, Raphaël Grandin, Jean-Michel Morel, *A Brief Evaluation of InSAR Phase Denoising and Coherence Estimation with Φ-Net*, Image Processing On Line, 14 (2024), pp. 205–216. https://doi.org/10.5201/ipol.2024.549

1 Introduction

Interferometric Synthetic Aperture Radar (InSAR) is a remote sensing technique that mainly measures ground topography or deformation. This is possible by combining two different Synthetic Aperture Radar (SAR) Single Look Complex (SLC) images z_1 and z_2 acquired at slightly different locations and/or dates. An interferogram Γ is computed by applying the complex multiplication

$$\Gamma = z_1 \cdot z_2^*, \tag{1}$$

where $*$ denotes the conjugation operation. The interferometric phase Φ_z is defined and interpreted by

$$\Phi_z = \angle \Gamma = \frac{-4\pi}{\lambda} \Delta R + \Phi_{noise}, \tag{2}$$

where $\angle$ refers to the phase of the complex number $\in [-\pi, \pi)$, λ is the radar carrier wavelength, and ΔR is the change of distance between the reflector and the satellite. Therefore, Φ_z is related to the change in distance to the target, which can be linked to the scene topography and deformation in the satellite's line of sight, among other factors. Since Φ_z can only be observed in a 2π interval, phase jumps called fringes will be present in the signal. Phase unwrapping is, therefore, a necessary processing step to obtain a consistent phase field. However, in practice, the observed phase is contaminated by noise Φ_{noise} due to several decorrelating effects (thermal noise, speckle, change in the scattering mechanism, different acquisition geometries, and processing errors, ...). This increases the difficulty of phase unwrapping. Therefore, phase denoising is usually performed before phase unwrapping. Furthermore, it is usually of interest to measure the local complex signal correlation γ_0 of the two SAR images defined as

$$\gamma_0 = \frac{\mathbb{E}\left(z_1 \cdot z_2^*\right)}{\sqrt{\mathbb{E}\left(|z_1|^2\right) \cdot \mathbb{E}\left(|z_2|^2\right)}} = \rho \cdot \mathrm{e}^{j\phi}. \tag{3}$$

Here, γ_0 can be decomposed into the coherence $\rho = |\gamma_0| \in [0, 1]$ and the denoised phase $\phi = \angle \gamma_0$. The coherence ρ is usually a good indicator of the phase noise level and, consequently, of the reliability of pixels for further analysis. For example, it can be used as a quality map in phase unwrapping. Coherence estimation can also be useful in other applications, such as change detection, where a local coherence loss could indicate that the area is affected by a change between the two dates.

1.1 Related Work

Algorithms for estimating the InSAR denoised phase and coherence have been actively developed throughout the years. Here, we will mention a few algorithms and refer to [14] for a more comprehensive overview. The most straightforward method is the multilook filter [7], i.e. a simple boxcar moving average filter chosen as a proxy of the expectation operation $\mathbb{E}$ in Equation (3). Its main drawback is that the estimation quality deteriorates when the signal is not locally stationary within the estimation window. This has motivated the introduction of adaptive filtering techniques such as the Lee filter in [8], where directional windows are used to consider the fringe direction. Also, local noise statistics are considered in the final estimation. On the other hand, it is also possible to filter the interferogram in a transformed domain. For example, a widely used filter in the frequency domain is known as the Goldstein filter [4]. This method denoises patches in the image separately and then recombines them to form the final denoised image. The filter is designed as the power α of the smoothed modulus of the Fourier transform of the patch (α varies between 0 and 1 and controls the filtering power). Similarly, sparse representations have been used to tackle this problem in the method termed SpInPhase [5], where a dictionary of patches is learned from the data, and each patch is represented as a sparse linear combination of dictionary patches. More recently, non-local patch-based filters have been applied to InSAR data because of their texture preservation properties. These methods start by assigning weights to patches in a local search window based on a similarity measure to the central patch. Then, the patches are typically used for joint filtering before recombining to provide the final filtered result. NL-InSAR [3] is a typical patch-based method characterized by its iterative scheme. At each iteration, similarity weights with neighboring patches are set based on the posterior probability of having the same underlying parameters. Then, a weighted maximum likelihood estimate is used to provide the filtered phase, coherence and reflectivity for the current iteration. Simply put, the similarity criterion and the estimate are based on the SAR data statistics. In [11], the BM3D [2] non-local collaborative filtering algorithm is extended to InSAR and denoted InSAR-BM3D. Then, in [10], an offset compensation criterion for patch similarity evaluation is established. Thus, OC-INSAR-BM3D is the offset-compensated version of InSAR-BM3D and demonstrates state-of-the-art filtering performance.

On the other hand, in recent years, Convolutional Neural Networks (CNNs) have been applied successfully to image processing tasks. In [15], the DnCNN network was proved to perform highly effective image denoising. Residual learning is highlighted as an important factor to boost denoising performance. In [6], the first CNN with residual connection is used for InSAR phase denoising. Φ-Net [12] combines residual learning with the well-known U-net architecture [9] to solve the interferometric phase denoising and coherence estimation task. Even though more recent articles tackling the problem of InSAR phase denoising using CNN architectures have emerged [13], Φ-Net [12] showed state-of-the-art performances at the time of its release. Furthermore, the model is available online for testing, which makes our evaluation study possible.

1.2 Paper Organization

In this work, we examine and test the network Φ-Net [12]. The article is organized as follows: in Section 2, we briefly inspect the method, the network architecture, the simulated dataset, and the training strategy. In Section 3, we perform experiments on some Sentinel-1 interferograms and compare the estimates of Φ-Net against the boxcar uniform filter. Section 4 provides a short guide for running the demo.

1.3 Mathematical Notation

In this section, we introduce the notation for uniform image filtering used in the article. Consider an image I, and denote $U_{M\times N}$ the uniform filter window centered at the origin with size $M \times N$. The uniform filtering operation $(U_{M\times N} \circledast I)(i,j)$ at spatial position (i,j) can be written as:

$$(U_{M\times N} \circledast I)(i,j) = \frac{1}{MN} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} \sum_{n=-\frac{N-1}{2}}^{\frac{N-1}{2}} I(i+m, j+n). \tag{4}$$

2 Method

This section will first explain the forward inference for estimating the noise-free parameters ρ and ϕ of a couple of Single Look Complex (SLC) images z_1 and z_2. Specifically, some pre-processing and post-processing are highlighted, as well as the network architecture. Then, we will briefly look at the training data and the training strategy used by the authors since we did not re-train the network. Note that the code for this part is not made public and that the following section is based solely on the original article [12].

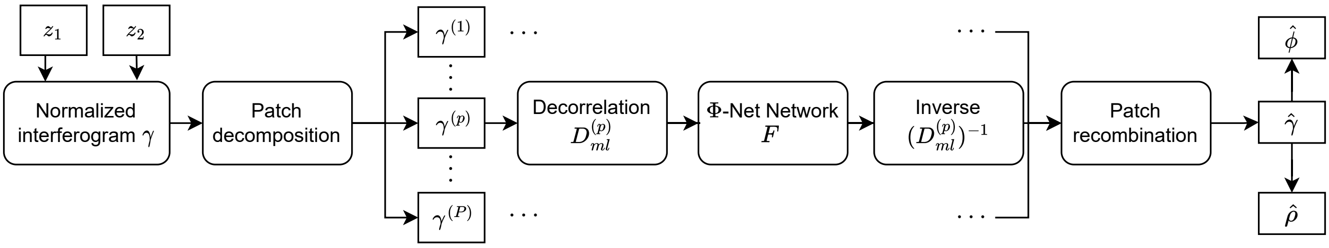


Figure 1: Denoising workflow with Φ-Net. The normalized interferogram is decomposed into P patches. Each patch is decorrelated and then fed to the network. The inverse transform is then applied, and the patches are recombined.

2.1 Forward Estimation

Starting from the two images z_1 and z_2, several steps are needed for the forward estimation as seen in Figure 1:

1. A normalized interferogram is computed. To do so, it is assumed that both images have the same amplitude A, and an estimate $\hat{A}$ is obtained using the maximum likelihood expression

$$\hat{A} = \sqrt{U_{w\times w} \circledast \left(\frac{|z_1|^2 + |z_2|^2}{2}\right)}. \tag{5}$$

   In simple words, the intensity $|z_i|^2$ ($i \in \{1,2\}$) is first averaged between the two dates, then a uniform $w \times w$ ($w = 3$) filter is applied following the notation defined in Equation (4), and the square root is finally taken. Then, the interferogram is normalized with the estimated amplitude

$$\gamma = \frac{\Gamma}{\hat{A}^2} = \gamma_R + j\gamma_I. \tag{6}$$

2. The interferogram image is decomposed into 64 × 64 overlapping patches with a stride 8 in both dimensions. The superscript $^{(p)}$ is used to designate quantities related to the patch p. Note that because of the patch decomposition and the fact that the stride is taken as 8, it is preferable to have an image size multiple of 8 in both dimensions; otherwise, some NaN pixels will appear at the border of the result. The number of NaN pixels is the remainder of the division of the image dimension by 8.

3. For each patch, the decorrelating Karhunen–Loève transform $D_{ml}^{(p)}$ is applied

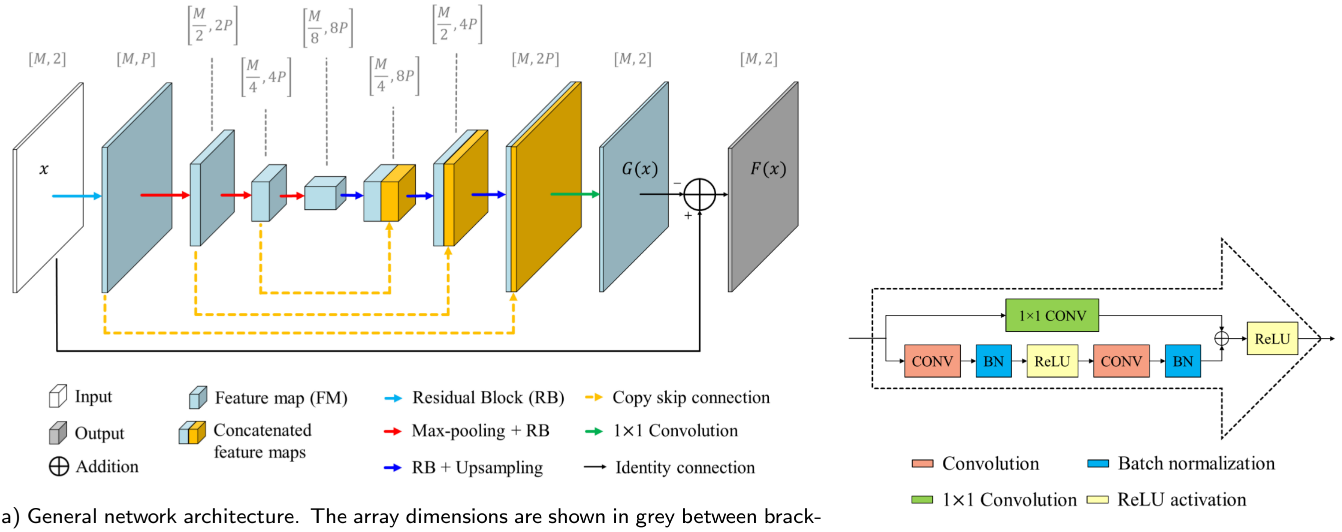
$$x^{(p)} = \begin{pmatrix} a^{(p)} \\ b^{(p)} \end{pmatrix} = D_{ml}^{(p)} \begin{pmatrix} \gamma_R^{(p)} \\ \gamma_I^{(p)} \end{pmatrix}, \tag{7}$$

   where $\gamma_R^{(p)}$ and $\gamma_I^{(p)}$ are the real and imaginary parts of $\gamma^{(p)}$ and $a^{(p)}$ and $b^{(p)}$ are the corresponding decorrelated quantities. The decorrelating matrix $D_{ml}^{(p)}$ has the following expression:

$$D_{ml}^{(p)} = \begin{pmatrix} \cos\phi_{ml}^{(p)} & \sin\phi_{ml}^{(p)} \\ -\sin\phi_{ml}^{(p)} & \cos\phi_{ml}^{(p)} \end{pmatrix}, \tag{8}$$

   where

$$\phi_{ml}^{(p)} = \angle \sum_{i=1}^{64}\sum_{j=1}^{64} \gamma^{(p)}(i,j)$$



(a) General network architecture. The array dimensions are shown in grey between brackets: [M, P] denotes a M x M array with P channels.

(b) Residual block architecture

Figure 2: Figures taken from the original article [12]. The network architecture is shown on the left. The residual block is shown on the right.

is the phase maximum likelihood estimate over the patch, i.e. it is a single scalar value for the whole patch. The matrix $D_{ml}^{(p)}$ is a rotation matrix with angle $-\phi_{ml}^{(p)}$ and can be interpreted as the matrix containing the eigenvectors of the noise covariance matrix (see [11] for more details).

4. $x^{(p)} = (a^{(p)}, b^{(p)})$ is fed to the network F as two separate channels, and a denoised estimate $F(x^{(p)}) = (\hat{a}^{(p)}, \hat{b}^{(p)})$ is obtained.

5. The inverse transform $(D_{ml}^{(p)})^{-1}$ is applied to the filtered patch

$$\begin{pmatrix} \hat{\gamma_R}^{(p)} \\ \hat{\gamma_I}^{(p)} \end{pmatrix} = (D_{ml}^{(p)})^{-1} \begin{pmatrix} \hat{a}^{(p)} \\ \hat{b}^{(p)} \end{pmatrix} . \tag{9}$$

6. The patches are recombined with averaging at overlapping positions. This gives an estimate of $\hat{\gamma} = \hat{\gamma}_R + j\hat{\gamma}_I$.

7. The denoised phase is taken as $\hat{\phi} = \angle\hat{\gamma}$ and the coherence as $\hat{\rho} = |\hat{\gamma}|$.

2.2 Network Architecture

The network architecture is shown in Figure 2. It is inspired by the U-Net network, with a shallower architecture since the input patches are small (M = 64). The first stage is an encoder, where the image size is successively reduced with 2×2 max pooling, whereas the number of channels increases. Since the parameter P that dictates the number of channels is set to P=64, the output of the encoder has a size $[\frac{M}{8}, \frac{M}{8}, 8P] = [8, 8, 512]$ (the size specified in the original article network description is $[4, 4, 512]$, which might indicate a small mistake regarding the array size). The second stage is the decoder, characterized by the inverse process, where the upsampling is conducted using bilinear interpolation. Skip connections are used to copy features (channels) from the encoding stage to the decoding stage.

In both the encoding and decoding stages, residual blocks are applied on the feature tensors instead of simple convolutions. Residual blocks that are part of operations represented as arrows in Figure 2a correspond to the detailed diagram of Figure 2b. Inside a residual block, the convolutional kernels are of size 3×3, except for the 1×1 residual connection. Also, notice that the residual blocks in Figure 2a always output kP channels, with $k \in \{1, 2, 4, 8\}$. In practice, for a residual block that outputs kP channels, all of its convolutions also output kP channels.

In Figure 2a, the last convolution is the only one not belonging to a residual block. It has a kernel of size 1×1, and it is used to get two channels representing the noise prediction $G(x)$. Then, the last residual connection is used to compute the denoised output $F(x)$ by combining the noise estimation $G(x)$ with the input x as $F(x) = x - G(x)$.

2.3 Dataset

2.3.1 Signal Model

The dataset is constructed via simulation of the phase, amplitude and coherence images. The objective is to have both noiseless (a_0, b_0) and noisy simulations (a, b) of the de-correlated channels. Under Goodman's model, z_1 and z_2 follow a zero mean circular Gaussian distribution with a covariance matrix given by

$$\begin{aligned}\Sigma &= \mathbb{E}\left\{\begin{pmatrix} z_1 \\ z_2 \end{pmatrix}\begin{pmatrix} z_1^* & z_2^* \end{pmatrix}\right\} \\ &= \begin{pmatrix} A^2 & A^2\rho e^{j\phi} \\ A^2\rho e^{-j\phi} & A^2 \end{pmatrix},\end{aligned} \tag{10}$$

where ρ and ϕ are the noiseless coherence and phase as seen in Equation (3), and A is the amplitude, considered equal for both images as previously mentioned.

Then, the model used for the simulation of the noisy images z_1 and z_2 from the underlying noiseless parameters A, ϕ, ρ is based on the following equation

$$\begin{aligned}\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} &= \mathrm{T}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \\ \mathrm{T} &= \begin{pmatrix} A & 0 \\ A\rho e^{-j\phi} & A\sqrt{1-\rho^2} \end{pmatrix},\end{aligned} \tag{11}$$

where u_1 and u_2 are two standard circular Gaussian random variables, and T is the Cholesky decomposition of the covariance matrix Σ. The simulation using Equation (11) is a simple way to ensure that the obtained data z_1 and z_2 follow a zero mean circular Gaussian distribution and that their covariance matrix equals $\Sigma = \mathrm{T}\mathrm{T}^H$, where T^H is the conjugate transpose of T. Then, the noisy (a, b) simulations are obtained from z_1 and z_2 by applying the normalization, patch decomposition, and decorrelation steps as shown in Figure 1. To get (a_0, b_0), the same decorrelation transform $D_{ml}^{(p)}$ should be applied to the noiseless $\gamma_0 = \rho e^{j\phi}$.

2.3.2 Noiseless Data Simulation

We need to simulate A, ρ and ϕ for the noiseless data simulation. First, For the amplitude and coherence, two types of simulations were used:

1. Linear horizontal Left-Right (LR) ramps or vertical Top-Bottom (TB) ramps,
2. Natural Patterns (NP) from optical images taken from the NWPU-RESISC45 dataset [1].

As for the phase, the SRTM (30 meters resolution) digital elevation model was used at two locations in Austria with Tandem-X acquisition parameters to generate topographic phase simulations. From the simulations, two subsets of 250 patches presenting Low-Frequency (LF) and High-Frequency (HF) spatial variations, respectively, were selected. These phase simulations were associated with different combinations of amplitude and coherence simulations (linear ramps or natural patterns). Also, another 100 low-frequency phase patches were selected and combined with amplitude and coherence images selected from natural patterns. In this case, the coherence images were segmented based on a watershed technique with two segmentation intervals: $0.6 < \rho < 0.8$ and $0.8 < \rho < 1$. In each segment, a random phase Step was added to the Low-Frequency pattern (LF+Step). The phase steps were simulated from a zero mean Gaussian distribution with standard deviation $\sigma = \pi\frac{\sqrt{2}}{6}$ which ensures that 3σ is in the $[-\pi, \pi)$ interval. Table 1 summarizes the combinations used for the noiseless simulation. In the end, 600 simulations of size 256 x 256 were obtained. These simulations reflect different patterns and correlations between the amplitude, coherence and phase.

Data	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Amplitude	LR	TB	NP	TB	NP	NP
Coherence	LR	LR	LR	NP	NP	NP
Phase	LF/HF	LF/HF	LF/HF	LF/HF	LF/HF	LF+Step

Table 1: The combination of different patterns for the noiseless data simulation. LR refers to Left-Right ramp, TB refers to Top-Bottom ramp, NP refers to Natural Patterns. LF/HF refers to 50 images of Low-Frequency fringes and 50 of High-Frequency fringes, while LF+Step refers to 100 images of Low-Frequency fringes on which random phase Steps were added in some segments based on the coherence. Each column is a simulation of 100 images, which gives a total of 600 images. The image size is 256×256.

2.4 Training

The loss used for the training is a combination of an L_2 term with a regularization term R as follows

$$\begin{aligned} L\left(a_0, b_0, \hat{a}, \hat{b}\right) &= L_2\left(a_0 - \hat{a}, b_0 - \hat{b}\right) + \lambda R(\hat{a}, \hat{b}) \\ &= \left\| \begin{bmatrix} a_0 - \hat{a} \\ b_0 - \hat{b} \end{bmatrix} \right\|_2 + \lambda \cdot \left\| \begin{bmatrix} \max(0, |\hat{a}| - 1) \\ \max(0, |\hat{b}| - 1) \end{bmatrix} \right\|_1 . \end{aligned} \tag{12}$$

The first term is an L_2 loss term between the predicted denoised channels $(\hat{a}, \hat{b})$ and the ground truth noiseless data (a_0, b_0). The second term is a regularization that encourages the output to be less than 1 in absolute value. It is only active when at least one of the outputs $(\hat{a}, \hat{b})$ is greater than 1 in absolute value, and in this case, the L_1 loss is used to penalize it. Parameter λ was used to adjust the regularization amount and was empirically set to $\lambda = 10^{-2}$.

The dataset was split into 540 training images and 60 validation images. The training data was also split into 64×64 patches of stride 8. Since the patches have a lot of overlap, to augment the diversity of the data, each patch was rotated by 90° with respect to the neighboring patch, and the sign was switched every four patches. This process yielded $N_T = 337408$ patches for training and $N_V = 37376$ patches for validation. The batch size was taken as $B_s = 128$. The Adam algorithm was utilized with an initial learning rate of $l_r = 10^{-4}$, which was subsequently divided by factors of 10, 20, and 30 after 15, 30, and 45 epochs, respectively. The training converged after 50 epochs, and the validation loss indicated no over-fitting.

3 Experiments

In this section, we conducted a few experiments with the Φ-Net network to assess its denoising capabilities.

We compared the results against the simple boxcar filter given by

$$\hat{\gamma}_{\text{box}} = \frac{U_{s\times s} \circledast (z_1 \cdot z_2^*)}{\sqrt{\left(U_{s\times s} \circledast |z_1|^2\right) \cdot \left(U_{s\times s} \circledast |z_2|^2\right)}} = \hat{\rho}_{\text{box}} \cdot \mathrm{e}^{j\hat{\phi}_{\text{box}}}, \tag{13}$$

where the expectation in Equation (3) is replaced by the empirical moving average of size $s \times s$ with the notation defined in Equation (4). We chose $s = 5$, as it is the same size selected in the Φ-Net article in its experimental section [12].

We conducted experiments using Sentinel-1 data on mountainous regions, for which the characteristics are listed in Table 2. Two aligned image crops c_1 and c_2 of size 256×256 were computed for each area of interest. The orbital phase contribution was also simulated as ϕ_{orb}, and compensated from the secondary image, such that

$$\begin{aligned} z_1 &= c_1, \\ z_2 &= c_2 \cdot e^{j\phi_{orb}}. \end{aligned} \tag{14}$$

After compensation, the interferogram Γ obtained through Equation (1) is mainly affected by topographic phase components, which is the most interesting contribution to observe on this terrain.

Idx	Area	Primary date	Secondary date	Centroid (lon, lat)	orbit	$B_\perp(m)$
A	Etna	20220411	20220505	14.9863, 37.7429	124	-139
B	El Capitan	20210710	20210728	-119.6358, 37.7422	144	-107
C	War Jan	20230118	20230130	65.3277, 32.4301	42	-310
D	Kilimanjaro	20180809	20180821	37.3586, -3.0648	79	130

Table 2: Sentinel-1 dataset consisting of image couples selected over mountainous regions at two dates with small temporal separation and big geometric baseline $B_\perp$ separation. This setting is designed to optimize the observation of topographic fringes.

The results are shown in Figure 3. Looking at the results globally, we can see that the Φ-Net phase and coherence are cleaner than the boxcar estimation. In particular, the fringes on the top of the image in example D are perfectly denoised with Φ-Net, whereas the boxcar filter struggles to completely remove the noise in this area. Notice that the fringes are dense and difficult to discern in the original noisy phase. Φ-Net also succeeds in estimating the coherence, attributing it a high value in this area, which indicates robustness to the fringe density.

Another observation can be made on the Φ-Net phase estimation in areas highly contaminated with noise (low coherence). At first glance, we might say that the network produces two different types of results in these regions. It either introduces high frequency fringes, such as the ones in A and D, or some low frequency smooth phase estimation. In fact, when looking at the boxcar results for the same areas, we notice a link with the Φ-Net results. When the boxcar is applied on completely noisy areas (top of image A), we see some small random color patches in the denoised phase. In this case, Φ-Net hallucinates fringes consistent with the boxcar pattern. Conversely, when the boxcar filter finds a common phase in the noisy region, i.e. the small patch colors are slightly coherent (bottom of B, or river in C), Φ-Net provides a filtered phase with this common color. This observation might be better interpreted if we look at the Φ-Net workflow in Figure 1. Indeed, the input image is decomposed into overlapping patches, and then the patch phase is subtracted (Decorrelation) before going into the network. Finally, the patch phase is added back, and the patches are recombined. Thus, the Decorrelation of patches is analogous to the boxcar estimation, and Φ-Net can be thought as being a network that extracts an additional signal from the residual of the boxcar filter. This might explain the visual link we see in the results of Φ-Net and the boxcar filter. In any case, even

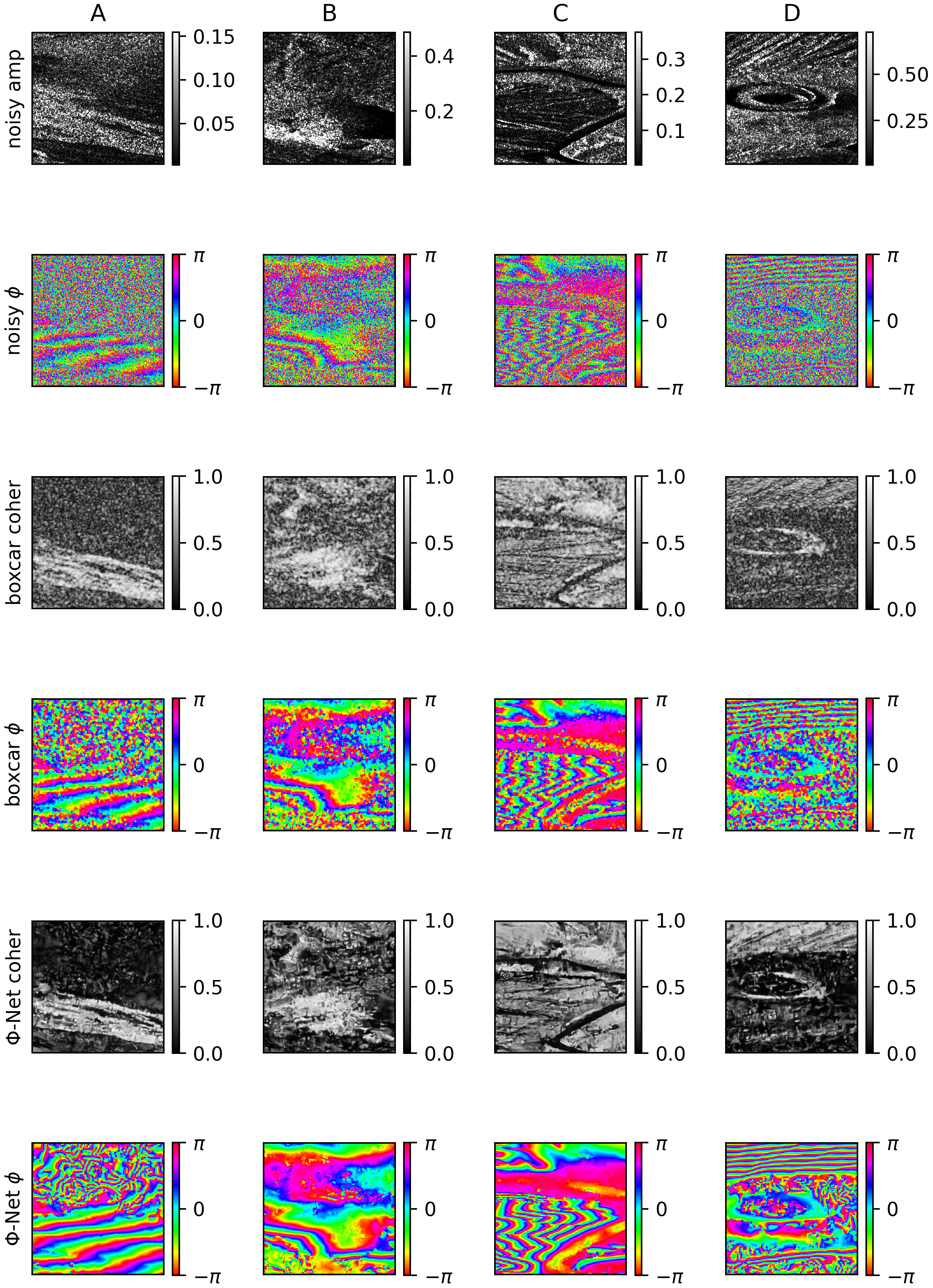


Figure 3: Performance comparison on the four areas of interest in Table 2. The first two rows show the noisy interferogram; then we show the denoised phase and coherence estimation with the boxcar filter and the Φ-Net network.

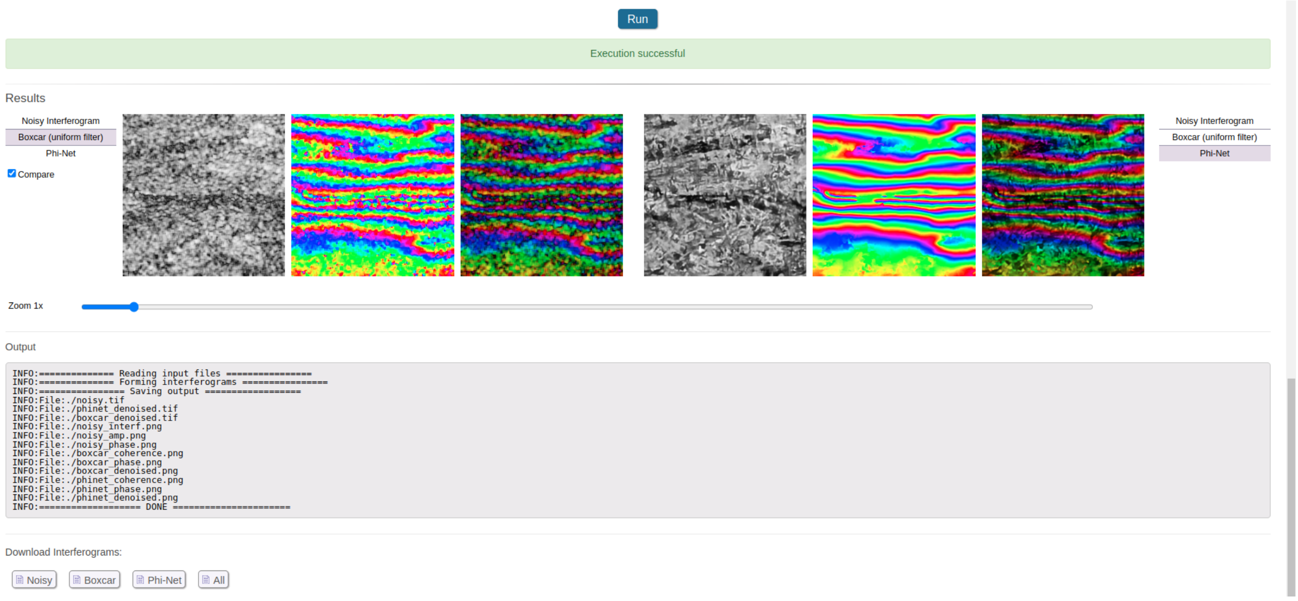
though introducing fringes in noisy areas might be considered undesirable, Φ-Net designates these areas as incoherent.

An interesting feature of Φ-Net is the very good resolution of the coherence estimation. For example, in image C, we can see the preservation of small details around the river. The river edges look very sharp, the coherent pixels from the river bank do not propagate to the river, i.e. the network is sure that the river area is incoherent and assigns it a low value. Other details that can be seen are small bright spots or linear structures. The width of such structures seems to be around 2 pixels, indicating only a small loss of resolution compared to the original Sentinel-1 image, which is impressive.

Our last observation is on image C, where some horizontal discontinuities can be seen very clearly in the Φ-Net phase fringes. These discontinuities can also be slightly seen in the boxcar results, which brings us back to the analogy between both methods. However, looking at the original interferogram, one could hope to recover fringes without these discontinuities. This might be attributed to the patch-based processing upon which Φ-Net is built. Perhaps using a fully convolutional neural network, one might increase the receptive field beyond the prescribed patch size (64 in this case) and recover globally consistent fringes.

4 Demo

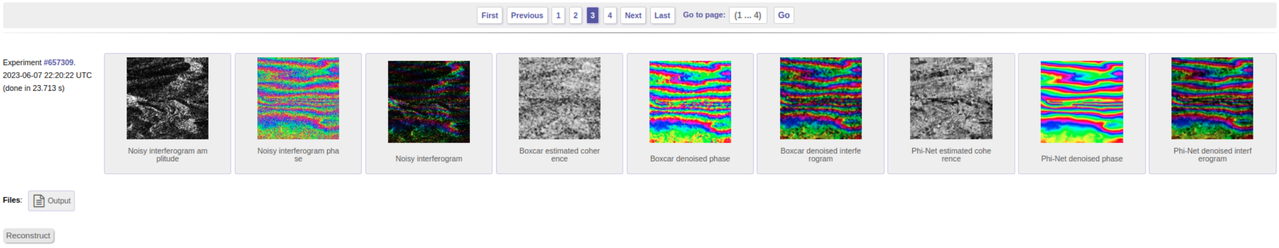
In this section, we illustrate the demo usage through an example. On the main page of the demo website, the user will find an exhaustive overview of the input and output description.



```
INFO:============== Reading input files ================
INFO:============== Forming interferograms ================
INFO:================ Saving output ==================
INFO:File:./noisy.tif
INFO:File:./phinet_denoised.tif
INFO:File:./boxcar_denoised.tif
INFO:File:./noisy_interf.png
INFO:File:./noisy_amp.png
INFO:File:./noisy_phase.png
INFO:File:./boxcar_coherence.png
INFO:File:./boxcar_phase.png
INFO:File:./boxcar_denoised.png
INFO:File:./phinet_coherence.png
INFO:File:./phinet_phase.png
INFO:File:./phinet_denoised.png
INFO:=================== DONE ======================
```

Figure 4: The interface of the demo result. The image gallery allows flipping between results and a side-by-side comparison as well. Messages are printed out to indicate the correct execution of the demo. The user can also click on the download buttons to obtain the images in TIFF format or, for the "All" button, download an archive containing all the images in ZIP format.

In particular, we highlight the image format, order, preprocessing, and size. The output order and colormap descriptions are also detailed. The demo does not have parameters, so the user only has to select the inputs by clicking on the "Upload data" button or picking any of the suggested examples. After clicking the "Run" button, the user has to wait until the execution completes. The typical runtime for 256 x 256 images is around 30 seconds. When the execution finishes, the results are displayed in a gallery, as seen in Figure 4. The plots of the results and the log text file are stored in the demo's archive (Figure 5) and can be revisited later, downloaded, or visualized with the "Reconstruct" button. However, the TIFF arrays are not archived, and attempting to download the files will result in an error.

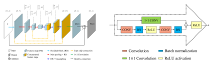


Experiment #657309.
2023-06-07 22:20:22 UTC
(done in 23.713 s)

First
Previous
1
2
3
4
Next
Last
Go to page:
(1 ... 4)
Go
Noisy interferogram amplitude
Noisy interferogram phase
Noisy interferogram
Boxcar estimated coherence
Boxcar denoised phase
Boxcar denoised interferogram
Phi-Net estimated coherence
Phi-Net denoised phase
Phi-Net denoised interferogram
Files:
Output
Reconstruct

Figure 5: The archive interface. All the plots are saved and the output gallery can be reconstructed by clicking on the "Reconstruct" button.

Image Credits

 Network architecture from [12]

References

[1] G. Cheng, J. Han, and X. Lu, *Remote Sensing Image Scene Classification: Benchmark and State of the Art*, Proceedings of the IEEE, 105 (2017), pp. 1865–1883, https://doi.org/10.1109/JPROC.2017.2675998.

[2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering*, IEEE Transactions on Image Processing, 16 (2007), pp. 2080–2095, https://doi.org/10.1109/TIP.2007.901238.

[3] C. A. Deledalle, L. Denis, and F. Tupin, *NL-InSAR: Nonlocal Interferogram Estimation*, IEEE Transactions on Geoscience and Remote Sensing, 49 (2011), pp. 1441–1452, https://doi.org/10.1109/TGRS.2010.2076376.

[4] R. M. Goldstein and C. L. Werner, *Radar Interferogram Filtering for Geophysical Applications*, Geophysical Research Letters, 25 (1998), pp. 4035–4038, https://doi.org/10.1029/1998GL900033.

[5] H. Hongxing, J. M. Bioucas-Dias, and V. Katkovnik, *Interferometric Phase Image Estimation Via Sparse Coding in the Complex Domain*, IEEE Transactions on Geoscience and Remote Sensing, 53 (2015), pp. 2587–2602, https://doi.org/10.1109/TGRS.2014.2361919.

[6] N. K. Kottayil, A. Zimmer, S. Mukherjee, X. Sun, P. Ghuman, and I. Cheng, *Accurate Pixel-Based Noise Estimation for InSAR Interferograms*, in IEEE Sensors, 2018, pp. 1–4, https://doi.org/10.1109/ICSENS.2018.8589665.

[7] J. S. Lee, A. R. Miller, and S. A. Mango, *Intensity and Phase Statistics of Multilook Polarimetric and Interferometric SAR Imagery*, IEEE Transactions on Geoscience and Remote Sensing, 32 (1994), pp. 1017–1028, https://doi.org/10.1109/36.312890.

[8] J.-S. Lee, K. Papathanassiou, T. Ainsworth, M. Grunes, and A. Reigber, *A New Technique for Noise Filtering of SAR Interferometric Phase Images*, IEEE Transactions on Geoscience and Remote Sensing, 36 (1998), pp. 1456–1465, https://doi.org/10.1109/36.718849.

[9] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, in Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, 2015, pp. 234–241, https://doi.org/10.1007/978-3-319-24574-4_28.

[10] F. Sica, D. Cozzolino, L. Verdoliva, and G. Poggi, *The Offset-Compensated Nonlocal Filtering of Interferometric Phase*, Remote Sensing, 10 (2018), https://doi.org/10.3390/rs10091359.

[11] F. Sica, D. Cozzolino, X. X. Zhu, L. Verdoliva, and G. Poggi, *InSAR-BM3D: A Nonlocal Filter for SAR Interferometric Phase Restoration*, IEEE Transactions on Geoscience and Remote Sensing, 56 (2018), pp. 3456–3467, https://doi.org/10.1109/TGRS.2018.2800087.

[12] F. Sica, G. Gobbi, P. Rizzoli, and L. Bruzzone, *Φ-Net: Deep Residual Learning for InSAR Parameters Estimation*, IEEE Transactions on Geoscience and Remote Sensing, 59 (2021), pp. 3917–3941, https://doi.org/10.1109/TGRS.2020.3020427.

[13] S. Vitale, G. Ferraioli, and V. Pascazio, *A CNN Based Solution for InSAR Phase Denoising*, in International Geoscience and Remote Sensing Symposium (IGARSS), IEEE, 2022, pp. 931–934, https://doi.org/10.1109/IGARSS46834.2022.9883956.

[14] G. Xu, Y. Gao, J. Li, and M. Xing, *InSAR Phase Denoising: A Review of Current Technologies and Future Directions*, IEEE Geoscience and Remote Sensing Magazine, 8 (2020), pp. 64–82, https://doi.org/10.1109/MGRS.2019.2955120.

[15] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, *Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising*, IEEE Transactions on Image Processing, 26 (2017), pp. 3142–3155, https://doi.org/10.1109/TIP.2017.2662206.