

Phase Linking, Explained

A conceptual guide to the `PhaseLinking`` operator (DS-InSAR / SqueeSAR) in the SNAP Microwave Toolbox.

Audience: users who want to understand **what the operator does, why it does it that way, and how to drive it** — not just which button to press. Read it start-to-finish, then follow the hands-on tutorial (`Phase-Linking-Tutorial.md`) for a step-by-step run. For the button-level reference see the in-app help (`PhaseLinkingOp.html`); for a runnable end-to-end walkthrough see the notebook `snap-nb-sar-ds-insar-timeseries.ipynb`.

1. The problem it solves

A repeat-pass interferogram carries ground motion in its **phase**. Between two acquisitions of the *same* patch of ground, the phase difference is (ideally) just geometry + deformation + atmosphere. That works beautifully on **point-like** targets — buildings, rocks, corner reflectors — whose scattering is stable pass to pass. This is classical **Persistent-Scatterer (PS)** InSAR.

Over **natural** surfaces — soil, vegetation, lava, debris, farmland — the scattering *rearranges* between passes. A single pair is **decorrelated**: its coherence is low and its phase is noise. Classical PS handles this by throwing those pixels away and keeping only the sparse point targets. On many scenes that discards most of the image.

Distributed-Scatterer (DS) processing instead *recovers* a usable phase history over those natural surfaces. The key realisation: a single noisy pair is weak, but a **stack** of N acquisitions gives you $N(N-1)/2$ pairwise interferograms of the same ground. If you estimate **one consistent phase per epoch** that best explains *all* those pairs jointly, the noise averages down and the surface becomes usable. That joint estimation is **phase linking**, and it is the engine inside every modern DS-InSAR tool (SqueeSAR, FRInGE, MiaplPy, dolphin/NISAR-ADT). `PhaseLinking` brings it natively into the SNAP Microwave Toolbox.

The payoff, concretely: on the Etna stack in the demo notebook, DS coherence on the volcano flanks roughly **doubles** ($\approx 0.33 \rightarrow \approx 0.62$) after phase linking — the difference between "unusable speckle" and "usable time-series target."

2. The core idea in one paragraph

For each output pixel, look at a small neighbourhood, keep only the neighbours that are *statistically the same kind of surface* as the centre (its **SHPs**), and use them as looks to estimate the $N \times N$ complex **coherence matrix** of the stack. That matrix encodes every pairwise coherence (magnitude) and every pairwise interferometric phase (argument). Phase linking then asks: *what single per-epoch phase vector φ best reproduces this whole matrix?* The answer is an eigen-problem. Finally, write each epoch's original amplitude back with its estimated phase — producing a "phase-linked" stack that is a **drop-in replacement** for the coregistered stack in every downstream operator.

3. How it works, step by step (as implemented)

This mirrors `PhaseLinkingOp.processPolarisation(...)` exactly.

3.1 SHP selection — "which neighbours are the same surface?"

Around each centre pixel the operator walks a search window (``windowAzimuth`` × ``windowRange``, default **21** × **7** pixels, forced odd). For every candidate in the window it runs a **two-sample test** comparing the candidate's amplitude time series against the centre's. Candidates that pass at significance ``shpAlpha`` (default **0.05**) are accepted as **Statistically Homogeneous Pixels (SHPs)** — pixels that behave like the same scatterer population.

Three tests are offered via ``shpTest``:

Test	Character	Used by
KS (Kolmogorov–Smirnov)	default; robust, distribution-free	SqueeSAR, FRInGE
AD (Anderson–Darling)	more powerful in the distribution tails	FRInGE
TLog (Welch t-test on log-amplitude)	cheapest, parametric	MiaplPy "fast" mode

The window is a **search radius, not a multilook box**: the point is to gather as many *genuinely homogeneous* looks as possible while refusing to average across a field boundary, a road, or a building. Bigger window → more potential SHPs (more looks, smoother estimate) but a higher risk of straddling more than one surface type.

3.2 Sample covariance → coherence matrix

From the accepted SHP set (call its size L), accumulate the $N \times N$ complex sample covariance and normalise it to the **coherence matrix** $\hat{\Gamma}$:

$$\begin{aligned}\hat{C}[i,j] &= (1/L) \sum_q s_i(q) \cdot \text{conj}(s_j(q)) \\ \hat{\Gamma}[i,j] &= \hat{C}[i,j] / \sqrt{\hat{C}[i,i] \cdot \hat{C}[j,j]}\end{aligned}$$

$|\hat{\Gamma}[i,j]|$ is the per-pair coherence; $\arg(\hat{\Gamma}[i,j])$ is the per-pair interferometric phase — both already "multilooked" over the SHP set. $\hat{\Gamma}$ is Hermitian.

Optional bias correction (`coherenceBiasCorrection`, default **off**): sample coherence magnitude is positively biased at low L . When enabled, each off-diagonal magnitude is debiased ($|\gamma|^2 \leftarrow \max(0, (L|\gamma|^2 - 1)/(L - 1))$), phase untouched. It **helps EVD** (down-weights noisy pairs) but can **hurt EMI** at low coherence, because EMI inverts the magnitude matrix and the correction injects variance there. Hence off by default; turn it on only with EVD.

3.3 Phase estimation — EVD vs EMI

Both estimators return a per-epoch phase vector ϕ from $\hat{\Gamma}$. This is the choice users ask about most, so here is the intuition, not just the formula.

EVD (Eigenvalue Decomposition) — estimator = EVD, the default. $\hat{\Gamma}$ under the ideal DS model is rank-1 ($\hat{\Gamma} \approx u u^H$), so its **dominant eigenvector** u is the phase history: $\phi_n = \arg(u_n)$. One Hermitian eigendecomposition

per pixel. Fast, robust, and the original SqueeSAR estimator (Ferretti 2011). Its weakness: it weights every pair **equally**, so a few very noisy long-temporal-baseline pairs pull the estimate around.

EMI (Eigenvalue-based Maximum-likelihood estimator of Interferometric phase) — estimator = EMI (Ansari–De Zan–Bamler 2018). EMI reads the phase from the **smallest**-eigenvalue eigenvector of

$$M = \Gamma^{-1} \circ \hat{T}, \quad \Gamma = |\hat{T}| \quad (\Gamma^{-1} = \text{matrix inverse of the magnitude matrix; } \circ = \text{Hadamard product})$$

The Γ^{-1} weighting **down-weights low-coherence (long-baseline) pairs** — exactly where EVD's equal weighting hurts. That is EMI's whole advantage: **lower bias and lower variance when coherence is low**, at the same $O(N^3)$ cost. Rule of thumb: **EVD is the safe default; switch to EMI on stacks with strong temporal decorrelation** (vegetation, long time span). Note Γ^{-1} is the true matrix inverse — *not* the elementwise reciprocal (an early draft got this wrong; the shipped code is correct and covered by a Cramér–Rao-bound test).

3.4 Temporal coherence — the quality band

After estimating ϕ , the operator scores **how well one consistent phase history explains the whole matrix** (Pepe & Lanari goodness-of-fit):

$$\gamma_T = (2 / N(N-1)) \cdot | \sum_{\{i < j\}} \exp(j \cdot (\arg \hat{T}[i, j] - (\phi_i - \phi_j))) |$$

$\gamma_T \in [0, 1]$: **1** = the model perfectly reproduces every pair; **~1/N** = noise. This is emitted as the `tempCoh` band (on by default) and is *your quality mask* — keep pixels above ~0.6–0.7. Pixels below `tempCohMin` (default **0.6**) are not trusted (see pass-through below). This is a per-pixel confidence measure computed from the data itself — exactly what you want for deciding which pixels to trust.

3.5 Reference epoch — the zero-phase datum

The eigenvector solution is only defined up to a global phase rotation. The operator fixes that gauge by declaring one epoch the **zero-phase datum** ($\phi_{\text{ref}} = 0$), so its band is written **real-valued**. Chosen via `referenceEpochDate` (ddMMMyyyy); empty $\rightarrow$ the **chronological median** epoch (matching SBASInversionOp, and the best-conditioned choice). This is downstream-invariant: every consumer uses only *relative* phases $\arg(s_i \cdot \text{conj}(s_j))$, in which the datum cancels. This is the single-reference convention of dolphin / MiaplPy / FRInGE.

3.6 Output stack — and the "never degrade the input" guarantee

For each epoch the operator writes back a complex sample with the **original amplitude** and the **estimated phase**: $s_n = |s_n| \cdot \exp(j\phi_n)$. Band names are re-tagged (`i_p1..._ref/_sec<n>...`) and the Reference_bands/Secondary_bands metadata rewritten, so the result is a **drop-in** for InterferogramOp, CoherenceOp, MultiMasterInSAROp, SBASInversionOp — no graph changes.

Three **safety gates** decide when a pixel is *not* phase-linked, and in every one the operator **passes the original SLC samples through unchanged** rather than emitting garbage — so the output stack is never worse than the input:

1. **Any invalid/nodata epoch** at that pixel $\rightarrow$ pass-through (`tempCoh = NaN`). The covariance assumption needs all epochs valid.
2. **Too few SHPs**: `shpCount < max(shpMin, N)` $\rightarrow$ pass-through. The floor is raised to the stack size N because an $N \times N$ covariance built from fewer than N looks is **rank-deficient** (singular) — EVD would still return

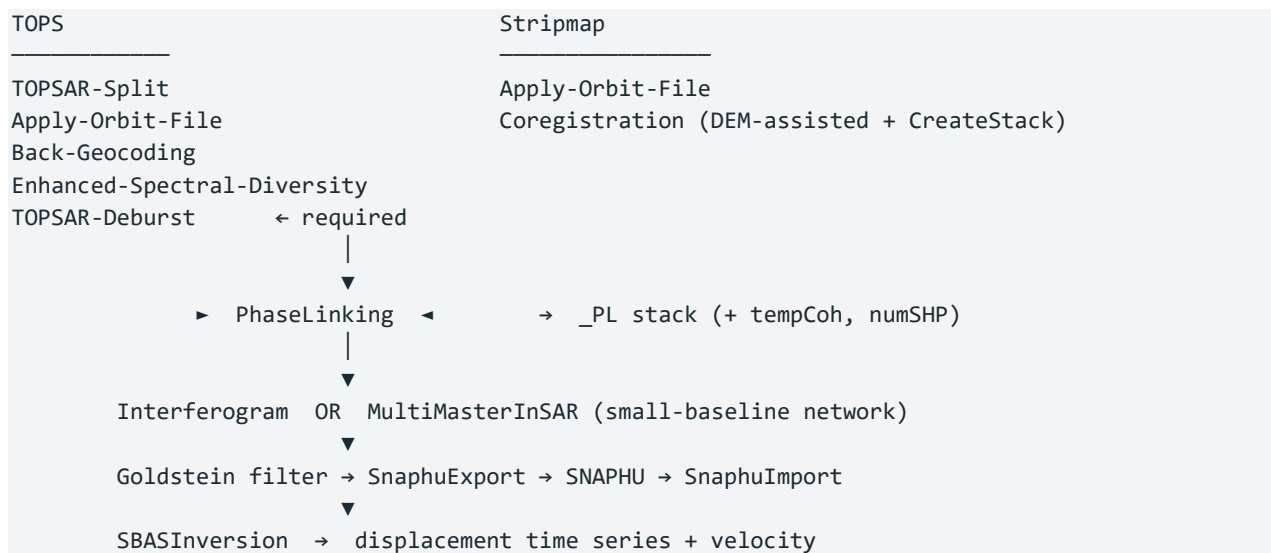
something, but EMI inverts $|\hat{\tau}|$ and would amplify null-space noise. A one-line info log fires when $\text{shpMin} < N$.

3. **Low temporal coherence:** $\gamma_T < \text{tempCohMin} \rightarrow$ pass-through, but tempCoh still records γ_T so you can mask on it.

This pass-through design is a deliberate correctness choice: phase linking only ever *improves* pixels it can improve, and leaves the rest exactly as the standard pipeline would see them.

Input requirements: a coregistered **complex** SLC stack with ≥ 3 epochs. TOPS input must be **deburst** first (TOPSAR-Deburst) — the operator throws on undebursted TOPS.

4. Where it sits in the pipeline



Phase linking changes **only phase** (amplitudes are preserved), so the downstream graph is identical to a standard pairwise-InSAR graph. **Skip it** if you only care about PS point targets; the gain is over *distributed* scatterers.

5. Parameter tuning — a practical guide

Parameter	Default	Raise / change it when...	Trade-off
windowAzimuth × windowRange	21 × 7	fields/homogeneous areas are large → bigger window = more looks = smoother, higher tempCoh	too big straddles multiple surfaces → biased estimate, blurred boundaries
shpTest	KS	AD for sharper tail discrimination; TLog when speed matters on huge stacks	AD/KS cost more than TLog
shpAlpha	0.05	lower (e.g. 0.01) =	fewer SHPs → more pass-

		stricter homogeneity, fewer but cleaner SHPs	through where scenes are heterogeneous
shpMin	20	raise for more conservative estimates	effective floor is always $\geq N$; too high $\rightarrow$ more pass-through
estimator	EVD	EMI on strongly decorrelating stacks (vegetation, long span)	EMI slightly more sensitive to magnitude noise; keep bias- correction off with it
referenceEpochDate	median	pin to a specific date only if you must align datums across runs	median is best- conditioned; leave empty normally
tempCohMin	0.6	raise (0.7–0.8) for a cleaner, sparser result; lower for denser but noisier	this is a masking threshold, not a processing knob — you can also re-threshold tempCoh afterwards
coherenceBiasCorrection	off	on with EVD on few-look scenes	can degrade EMI — do not combine

First-run recipe: defaults, EVD, KS, and *inspect the `tempCoh` band*. If large homogeneous areas come back with low tempCoh, enlarge the window; if you see strong decorrelation, try EMI; enable numSHP (outputShpCount) to confirm you are actually gathering enough looks.

6. What v1 does not do (current limitations)

- **Single-shot, full-stack.** Recommended operating range $N \leq \sim 50$ epochs. Long stacks ($N > 100$) want *sequential ministack + compressed SLC* processing (dolphin/FRInGE style) for conditioning and to bound temporal decorrelation — **deferred to v2**.
- **No iterative MLE / phase-triangulation refinement** (SqueeSAR inner loop, PTA). EVD/EMI single-shot only — v2.
- **TOPS must be deburst upstream.** No per-burst SHP selection yet.
- **No persistent-scatterer detection.** A companion PSCandidateOp reusing the amplitude statistics is planned; today, DS only.
- **Fixed (non-adaptive) SHP window.** No NLSAR/AML adaptive multilooking.

These are scope boundaries, not defects — knowing them keeps your expectations aligned with what the operator delivers today.

7. Validation evidence

The algorithm ships with an executable test pyramid — 30 @Test methods across the phase-linking suite (TestPhaseLinkingOp 9, TestEstimators 9, TestSelectors 5, TestCovarianceMatrix 4, TestHermitianEigSolver 3):

- **Exact recovery** on synthetic rank-1 stacks with known phase, across coherence levels.
- **EMI-beats-EVD** under decaying coherence (Monte-Carlo), confirming the estimator's advantage empirically.
- **Cramér–Rao bound check** — the test asserts the EVD/EMI RMS-to-CRB ratio stays in **[0.75, 1.6]×** the theoretical lower bound (i.e. near-optimal, not blown up).
- **SHP-test calibration** — the AD critical values are set to the Scholz–Stephens constants (the same ones `scipy.stats.anderson_ksamp` uses); the selector test verifies the false-positive rate and detection power at α , not the constants themselves.
- **End-to-end synthetic DS recovery**, plus **datum-invariance** and **rank-guard pass-through** tests.

The correctness fixes made during development (EMI rewritten to the canonical Ansari form; AD standardisation; temporal-coherence sign convention; rank-deficiency guard) are each pinned by one of these tests. Still on the roadmap: a **real-S1-vs-MiaplPy benchmark** on a common stack — the natural next validation step.

8. Frequently asked questions

- **"Is this the same algorithm as SqueeSAR/MiaplPy/dolphin?"** Yes — same covariance-based joint phase estimation. EVD = the SqueeSAR estimator; EMI = the Ansari 2018 estimator that dolphin/MiaplPy default to. Both are available.
- **"Can it make results worse than the standard pipeline?"** No — by design. Every pixel it cannot confidently link is *passed through unchanged*, so a phase-linked interferogram is never worse than the raw one over the same pixel; it is better over DS pixels.
- **"How do I know which pixels to trust?"** The `tempCoh` band — a data-driven per-pixel goodness-of-fit in `[0,1]`. Mask on it.
- **"EVD or EMI?"** EVD by default; EMI when coherence is low. Not bias-correction + EMI together.
- **"Why must I deburst TOPS first?"** v1 works on a continuous deburst grid; per-burst SHP selection is v2.
- **"How big a stack?"** Up to ~50 epochs comfortably; beyond that, ministack mode (v2).

9. References

- Ferretti, A. *et al.* (2011). *A new algorithm for processing interferometric data-stacks: SqueeSAR*. IEEE TGRS 49(9), 3460–3470.
- Ansari, H., De Zan, F., Bamler, R. (2018). *Efficient phase estimation for interferogram stacks*. IEEE TGRS 56(7), 4109–4125. (EMI)
- Cao, N., Lee, H., Jung, H.-C. (2016). *Mathematical framework for phase-triangulation algorithms in distributed-scatterer interferometry*. IEEE GRSL.

- Mirzaee, S., Amelung, F., Fattahi, H. (2023). *Non-linear phase linking using joined distributed and persistent scatterers*. Computers & Geosciences 171. (*MiaPlPy*)
- ISCE-framework / **dolphin** — reference phase-linking implementation used by the NISAR ADT:
<https://github.com/isce-framework/dolphin>

Companion documents: in-app help `PhaseLinkingOp.html` (parameter reference) · `snap-nb-sar-ds-insar-timeseries.ipynb` (runnable tutorial) · `Phase-Linking-Op-Spec.md` (implementation contract).