Corner Reflector راهنمای جامع پردازش واقعی

GECORIS و Sentinel-1، SNAP با

CR-InSAR و RCS، SCR، ALE از دریافت داده تا

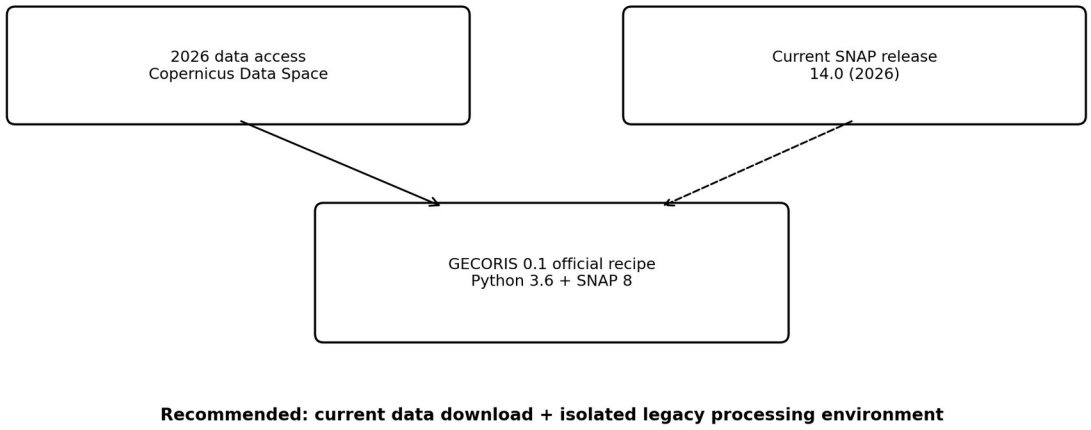
نسخه اجرایی - سپتامبر ۲۰۲۶

مبنای فنی: مستندات رسمی GECORIS 0.1، مقاله Czikhardt et al. (2021) و مستندات فعلی Copernicus/SNAP

نحوه استفاده از این راهنما

این سند یک ترجمه تحت‌اللفظی از مستندات نیست؛ بلکه یک دستور کار اجرایی است که ساختار رسمی GECORIS را با وضعیت واقعی دسترسی به داده‌ها در سال ۲۰۲۶ ترکیب می‌کند. هدف این است که بتوانید یک پروژه Corner Reflector را از صفر راه‌اندازی کنید، در هر مرحله بدانید چه فایل و خروجی باید ببینید و در صورت خطا بدانید کجا را بررسی کنید.

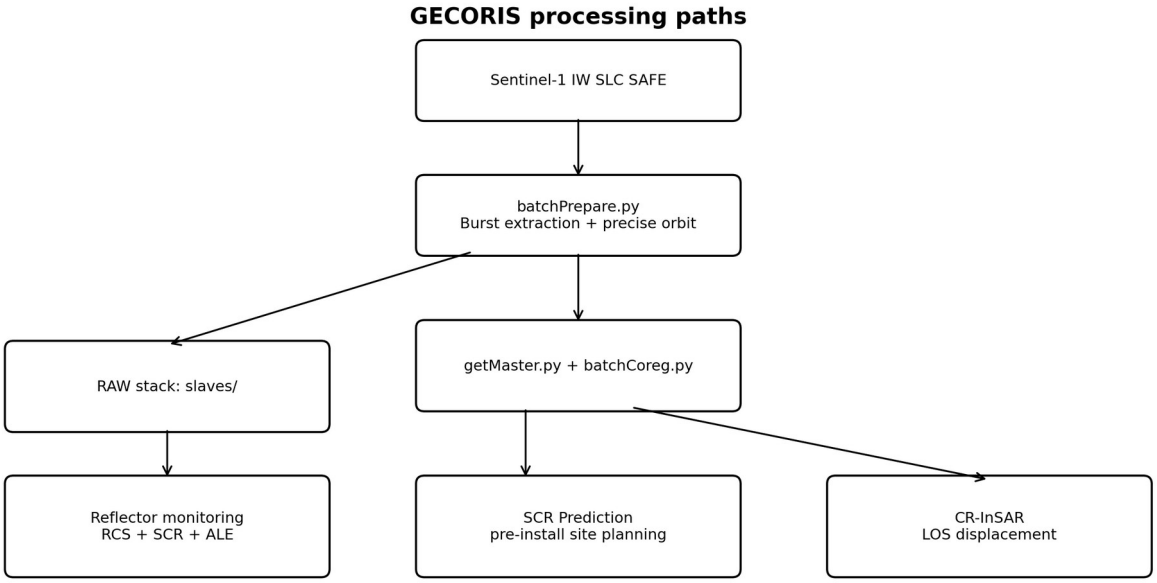
مهم‌ترین نکته سازگاری — مستندات رسمی GECORIS هنوز نسخه 0.1 را با Python 3.6 و SNAP 8 معرفی می‌کنند؛ در حالی که نسخه جاری SNAP در ۲۱ سپتامبر ۲۰۲۶ نسخه 14.0 است. بنابراین نصب «آخرین Python + آخرین SNAP» و اجرای کد قدیمی بدون تغییر، مسیر قابل اتکایی نیست. پیشنهاد این راهنما: داده را از سرویس فعلی Copernicus بگیرید، اما پردازش GECORIS را در یک محیط Legacy جداگانه و قابل بازتولید انجام دهید.



شکل ۱ - معماری پیشنهادی برای ترکیب دسترسی داده در ۲۰۲۶ با محیط پردازش سازگار با **GECORIS**.

نقشه راه کل پردازش

هدف	ورودی لازم	نیاز به **Co-registration**	خروجی اصلی
پایش عملکرد CR بعد از نصب	SLC burstهای استخراج‌شده در master/slaves	خیر	RCS، SCR، ALE و نمودارهای زمانی
انتخاب محل قبل از نصب	Stack هم‌مرجع Coregistered	بله	نقشه SCR GeoTIFF پیش‌بینی‌شده
پایش تغییرمکان CR با InSAR	Stack هم‌مرجع Coregistered + شبکه PS	بله	سری زمانی LOS و در چند Track تجزیه مولفه‌ها



شکل ۲ - سه شاخه اصلی پردازش **GECORIS** و نقطه‌ای که مسیر خام و **Coregistered** از هم جدا می‌شوند.

فهرست فصل‌ها

- ۱. مفاهیم ضروری SAR و Corner Reflector
- ۲. انتخاب سخت‌افزار و معماری نصب نرم‌افزار
- ۳. نصب محیط Legacy سازگار با GECORIS
- ۴. دریافت Sentinel-1 SLC در سال ۲۰۲۶
- ۵. سازمان‌دهی پروژه و کنترل کیفیت داده خام
- ۶. ساخت snap.parms و اجرای batchPrepare
- ۷. انتخاب Master و Co-registration
- ۸. ساخت reflectors.json و stacks.json
- ۹. SCR Prediction قبل از نصب
- ۱۰. Reflector Monitoring بعد از نصب
- ۱۱. تفسیر RCS، SCR و Outlierها
- ۱۲. مکان‌یابی دقیق و ALE
- ۱۳. CR-InSAR و سری زمانی تغییرمکان
- ۱۴. کنترل کیفیت نهایی و چک‌لیست تحویل
- ۱۵. خطاهای رایج و عیب‌یابی
- پیوست A. فایل‌های نمونه کامل

- پیوست B. دستورات سریع
- پیوست C. منابع

فصل ۱ - مفاهیم ضروری قبل از شروع پردازش

۱-۱. چرا **SLC** و نه **GRD**؟

GECORIS بر پایه اندازه‌گیری‌های Sentinel-1 SLC کار می‌کند. در SLC هر پیکسل یک مقدار مختلط دارد و دامنه و فاز را نگه می‌دارد. فاز برای InSAR ضروری است و ساختار نمونه‌برداری Range/Azimuth نیز برای مکان‌یابی زیرپیکسلی Corner Reflector حفظ می‌شود. GRD پس از تبدیل به Ground Range و آشکارسازی شدت، اطلاعات فاز مناسب InSAR را ندارد؛ بنابراین برای این Workflow انتخاب درستی نیست.

قاعده — برای پروژه GECORIS، Product Type را IW SLC انتخاب کنید. فایل را قبل از ورود به GECORIS به GRD، Terrain Corrected یا تصویر نقشه‌ای تبدیل نکنید.

۱-۲. **Subswath**، **IW** و **Burst**

حالت Interferometric Wide Swath یا IW در Sentinel-1 از سه Subswath با نام‌های IW1، IW2 و IW3 تشکیل می‌شود و در TOPS هر Subswath به Burstها تقسیم می‌شود. Corner Reflector در هر Track باید داخل Burst مشخصی قرار گیرد. تابع batchPrepare در GECORIS برای استخراج Burst متناظر با AOI طراحی شده است؛ بنابراین در حالت عادی نیازی نیست ابتدا به‌صورت دستی TOPSAR Split انجام دهید.

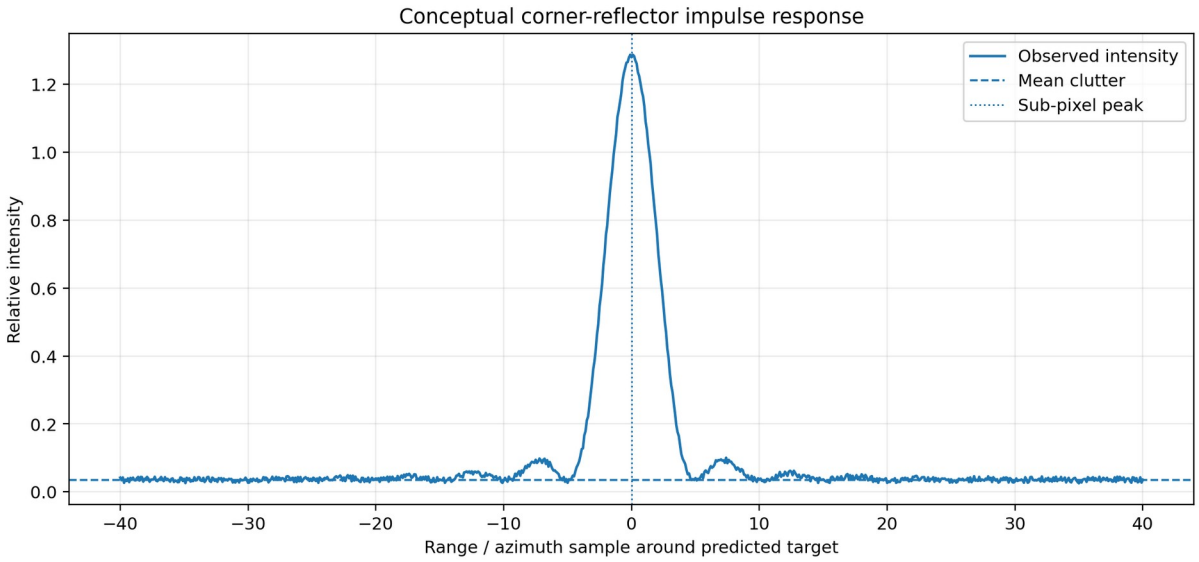
۱-۳. **Stack** در **GECORIS** یعنی چه؟

Stack در مستندات GECORIS به سری زمانی SLC مربوط به یک Orbit Track واحد گفته می‌شود. Ascending و Descending و Relative Orbitهای متفاوت را در یک Stack مخلوط نکنید. اگر یک CR توسط سه Track دیده می‌شود، سه Stack مستقل تعریف کنید و در مرحله تحلیل شبکه‌ای آن‌ها را کنار هم قرار دهید.

۱-۴. **Clutter**، **RCS** و **SCR**

Radar Cross Section یا RCS بیانگر قدرت بازتاب راداری هدف است. برای یک Corner Reflector ایده‌آل، RCS به هندسه، ابعاد، جهت‌گیری نسبت به رادار، جنس و طول موج وابسته است. رابطه عمومی مورد استفاده در مقاله GECORIS به صورت $RCS_CR = (4\pi/\lambda^2) \times A_eq^2$ بیان شده است که A_eq سطح معادل عمود بر موج است.

Clutter مجموع بازتاب‌های ناخواسته محیط پیرامون در همان سلول تفکیک است. SCR نسبت قدرت بازتاب CR به Clutter پیرامون است. به‌طور مفهومی هرچه SCR بزرگ‌تر باشد، تعیین Peak، فاز و مکان هدف پایدارتر است. GECORIS از SCR نه فقط به‌عنوان یک شاخص کیفیت، بلکه برای برآورد دقت مکان‌یابی و واریانس فاز InSAR نیز استفاده می‌کند.



شکل ۳ - نمایش مفهومی **Peak** پاسخ **Corner Reflector** روی زمینه **Clutter** و محل تعیین **Peak** زیرپیکسلی.

۱-۵. **ALE / APE** و **Peak** زیرپیکسلی

مختصات ژئودتیکی دقیق CR با مدار دقیق ماهواره و معادلات Range-Doppler به مختصات مورد انتظار در Radar Datum تبدیل می‌شود. سپس Peak واقعی پاسخ در SLC، با Oversampling FFT، در سطح Sub-pixel تعیین می‌شود. اختلاف مختصات مشاهده‌شده و پیش‌بینی‌شده در راستای Azimuth و Range اساس Absolute Location/Positioning Error است.

۱-۶. تفاوت سه هدف عملی

مسیر درست	اگر هدف شما این است...
RAW monitoring روی slaves/؛ Co-registration نیست لازم.	فقط بفهمم CR سالم است و خوب دیده می‌شود
ابتدا Coregistration، سپس scrSimul.py.	قبل از نصب، محل با Clutter کم پیدا کنم
Coregistration + پردازش InSAR + استخراج Time Series.	تغییرمکان میلی‌متری CR را محاسبه کنم

فصل ۲ - معماری سیستم و انتخاب محیط اجرای واقعی

۱-۲. وضعیت نسخه‌ها در ۲۰۲۶

صفحه نصب رسمی GECORIS حداقل Python 3.6 و SNAP 8.0 را ذکر می‌کند و اتصال SNAP به Python را با snappy-conf توضیح می‌دهد. در مقابل، صفحه رسمی STEP در سپتامبر ۲۰۲۶ SNAP 14.0 را نسخه جاری معرفی می‌کند. کد GECORIS پژوهشی است و مستندات رسمی سازگاری Plug-and-play با SNAP 14 را تضمین نمی‌کنند.

پیشنهاد اجرایی — برای اولین اجرای موفق، یک VM یا ماشین Linux جداگانه برای نسخه Legacy بسازید و آن را پس از راه‌اندازی Freeze/Backup کنید. سیستم روزمره خود را مجبور به Downgrade نکنید.

۲-۲. سخت‌افزار پیشنهادی

مولفه	حداقل برای تست	پیشنهادی برای سری زمانی
RAM	16 GB	32 تا 64 GB
CPU	4 هسته	8 تا 16 هسته
فضای دیسک	100 GB	500 GB تا چند TB، بسته به تعداد Track و SLC
Storage	SSD	NVMe SSD برای workDir
سیستم عامل	Linux/VM	Linux اختصاصی یا VM پایدار

اعداد بالا الزامات رسمی GECORIS نیستند؛ توصیه عملی برای جلوگیری از گلوگاه در SNAP هستند. در snap.parms مقدار ram و cores را متناسب با سخت‌افزار واقعی تنظیم کنید و کل RAM سیستم را به SNAP اختصاص ندهید.

۲-۳. معماری پیشنهادی پوشه‌ها

ساختار پیشنهادی پروژه

```
/data/CR_PROJECT/
├── raw_s1/                 # Full Sentinel-1 IW SLC SAFE / ZIP
├── tracks/
│   ├── DSC124/
│   │   ├── slaves/         # output of batchPrepare
│   │   ├── master/         # selected master
│   │   └── coreg/          # co-registered products (if needed)
│   └── ASC175/
├── config/
│   ├── snap_DSC124.parms
│   ├── reflectors.json
│   ├── stacks.json
│   ├── gecoris.parms
│   └── planning_DSC124.json
└── output/
    ├── monitoring/
    ├── scr_prediction/
    └── insar/
```

فصل ۳ - نصب محیط **Legacy** سازگار با **GECORIS**

۳-۱. نصب **Conda** و ساخت **Environment**

دستورهای زیر همان منطق نسخه رسمی را دنبال می‌کنند. Python 3.6 پایان عمر شده است؛ بنابراین ممکن است Conda جدید در حل وابستگی‌ها مشکل داشته باشد. اگر این اتفاق افتاد، از یک Miniconda قدیمی/VM آرشیوی یا محیط از قبل ساخته‌شده استفاده کنید. هدف اصلی بازتولید محیطی است که snappy مربوط به SNAP 8 بتواند با آن کار کند.

دستور رسمی وابستگی‌های **Python**

```
conda create -n gecoris python=3.6
conda activate gecoris
```

```
conda install numpy scipy matplotlib pandas shapely requests scikit-image h5py=2.10
conda install -c conda-forge gdal netcdf4
```

۳-۲. وابستگی‌های **Linux**

```
sudo apt-get update
sudo apt-get install libgfortran5 python3-distutils
# Official docs also mention libgfortran3 for the legacy stack.
# On newer Ubuntu versions it is no longer in the normal repository.
```

هشدار — libgfortran3 یکی از نقاط شکست رایج در نصب جدید است. به‌جای دستکاری خطرناک کتابخانه‌های سیستم اصلی، اجرای GECORIS را در VM/Container/سیستم Legacy جداگانه نگه دارید.

۳-۳. نصب **SNAP 8** و **Sentinel-1 Toolbox**

در Installer مربوط به SNAP 8، Microwave/Sentinel-1 Toolbox باید نصب باشد. پس از نصب، یک بار SNAP GUI را اجرا کنید تا ماژول‌ها و تنظیمات اولیه ایجاد شوند.

۳-۴. اتصال **SNAP** به **Python**

```
conda activate gecoris
which python
# Example output: /home/user/miniconda3/envs/gecoris/bin/python

cd /path/to/snap/bin
./snappy-conf /home/user/miniconda3/envs/gecoris/bin/python

# Then install the generated snappy package according to the SNAP 8 instructions.
cd $HOME/.snap/snap-python/snappy
python setup.py install
```

۳-۵. تست **snappy**

```
python - <<'PY'
import snappy
print('snappy import OK')
PY
```

Checkpoint 1 — تا وقتی import snappy بدون خطا اجرا نشده، وارد نصب GECORIS نشوید. خطای Java، jpy یا snappy باید همین‌جا حل شود.

۳-۶. دریافت کد **GECORIS**

```
cd ~/sw
git clone https://memorid@bitbucket.org/memorid/gecoris.git
```

```
# Add to ~/.bashrc
export gecoris_HOME=~/sw/gecoris
export PYTHONPATH=${PYTHONPATH}:${gecoris_HOME}
export PATH=${PATH}:${gecoris_HOME}/gecoris

source ~/.bashrc
```

امنیت فایل‌های .**parms** — کدهای این خانواده فایل پارامتر را با eval می‌خوانند. بنابراین فایل .parms ناشناس یا دریافت‌شده از منبع غیرقابل اعتماد را اجرا نکنید؛ از نظر Python می‌تواند حاوی Expression مخرب باشد.

فصل ۴ - دریافت Sentinel-1 SLC در سال ۲۰۲۶

۴-۱. منبع فعلی داده

Copernicus Data Space Ecosystem منبع فعلی دسترسی به Sentinel-1 است. این سرویس SLC کامل و همچنین SLC Burst را ارائه می‌دهد. چون GECORIS 0.1 برای Full SLC و استخراج Burst با SNAP نوشته شده است، برای اولین Pipeline از محصول کامل IW_SLC__1S در قالب SAFE استفاده کنید.

۴-۲. فیلترهای جستجو

فیلتر	مقدار پیشنهادی
Mission	Sentinel-1
Product Type	SLC
Acquisition Mode	IW
Polarization	VV یا محصول Dual که VV را دارد
Orbit Direction	Ascending یا Descending؛ جداگانه
Relative Orbit	برای هر Stack ثابت
Date range	ترجیحاً شامل چند تصویر قبل از نصب و تصاویر بعد از نصب

۴-۳. چرا داده قبل از نصب مهم است؟

برای جداسازی Clutter طبیعی سایت از پاسخ CR و برای ارزیابی افزایش RCS پس از نصب، داشتن Acquisitionهای پیش از نصب بسیار مفید است. در Station Log نیز installDate و startDate ثبت می‌شوند. اگر فقط داده بعد از نصب دارید، بخشی از تحلیل زمینه‌ای محدود می‌شود.

۴-۴. Sentinel-1C

Copernicus اعلام کرده داده Sentinel-1C از ژانویه ۲۰۲۵ در Data Space در دسترس است. GECORIS اصلی در ۲۰۲۱ نوشته شده و سازگاری آن با متادیتای S1C در مستندات قدیمی تصریح نشده است. بنابراین برای اولین Validation، در صورت امکان ابتدا با S1A Pipeline را تثبیت کنید و سپس S1C را روی چند Acquisition آزمایشی وارد کنید.

۵-۴. کنترل نام و محتوای محصول

- فایل باید SLC باشد، نه GRD.
- Acquisition Mode باید IW باشد.
- برای هر Stack، Orbit Direction و Relative Orbit باید یکسان بماند.
- محدوده زمانی باید CR و AOI را پوشش دهد.
- ZIP/SAFE را خراب یا ناقص نگه ندارید؛ یک محصول ناقص می‌تواند batchPrepare را در میانه کار متوقف کند.

Checkpoint 2 — حداقل یک SLC را در SNAP GUI باز کنید و در Product Explorer وجود IW1/IW2/IW3 و باند Complex مربوط به Polarization موردنظر را بررسی کنید.

فصل ۵ - سازمان‌دهی **Track** و تعیین **AOI/Subswath**

۵-۱. هر **Relative Orbit** یک پروژه پردازشی مستقل

اگر محل شما در DSC124 و ASC175 دیده می‌شود، دو workDir مستقل بسازید. نام Stack را کوتاه و ثابت نگه دارید، مثلاً s1_dsc124 و s1_asc175. این ID بعداً در JSON، plots و InSAR استفاده می‌شود.

۵-۲. **AOI** چقدر کوچک باشد؟

برای پایش یک یا چند CR، AOI را آن‌قدر کوچک بگیرید که فقط Burst لازم استخراج شود، اما آن‌قدر تنگ نباشد که خطاهای کوچک مختصات یا CRهای دیگر بیرون بیفتند. برای اولین تست می‌توانید Bounding Box چند کیلومتری اطراف شبکه بگیرید و پس از اطمینان آن را کوچک‌تر کنید. پارامترهای min_lon/max_lon/min_lat/max_lat در snap.parms فقط برای انتخاب پوشش مناسب استفاده می‌شوند.

۵-۳. تعیین **Subswath**

در صورت اطمینان از Subswath، فقط همان IW را وارد کنید. اگر AOI روی مرز Subswathها است، مستندات اجازه می‌دهند چند swath را به صورت Tuple تعریف کنید. این تصمیم را با Product Explorer در SNAP و Coverage واقعی کنترل کنید.

فصل ۶ - ساخت snap.parms و اجرای Burst Preparation

۶-۱. ساختار واقعی snap.parms

فایل snap.parms در مثال رسمی JSON استاندارد نیست؛ یک Dictionary با Syntax پایتون است. Tupleها با پرانتز نوشته می‌شوند و Pathها بهتر است مطلق باشند.

نمونه **snap.parms** - مقادیر مختصات و مسیرها را با پروژه خود جایگزین کنید

```
{
"SNAP_parameters" : {
    "snapPath" : "/home/user/snap/bin/",
    "repoPath" : "/home/user/sw/gecoris/gecoris/",
    "ram" : 24000,
    "cores" : 8
},
"Stack_parameters" : {
    "workDir" : "/data/CR_PROJECT/tracks/DSC124/",
    "dataDirs" : ("/data/CR_PROJECT/raw_s1/DSC124/",),
    "swaths" : ("IW1",),
    "min_lon" : "49.90",
    "max_lon" : "50.10",
    "min_lat" : "36.15",
    "max_lat" : "36.35",
    "startDate" : "20260101",
    "endDate" : "",
    "masterDate" : ""
}}
```

۶-۲. معنی پارامترها

پارامتر	معنی	نکته کنترلی
snapPath	مسیر Binaryهای SNAP	پوشه‌ای که gpt در آن قرار دارد
repoPath	مسیر package کد GECORIS	به پوشه gecoris داخل repo اشاره کند
ram	حافظه برای پردازش SNAP بر حسب MB	کمتر از RAM واقعی سیستم بگذارید
cores	تعداد Core	از کل هسته‌ها کمی کمتر
workDir	مسیر خروجی Track	برای هر Track جدا
dataDirs	پوشه‌های SLC	حتماً Tuple؛ حتی یک مسیر با کامای انتهایی
swaths	IWهای مورد استفاده	مثلاً ("IW1",)
startDate/endDate	فیلتر Acquisition	YYYYMMDD؛ endDate خالی یعنی ادامه تا آخر
masterDate	Master دستی	خالی یعنی انتخاب بعدی/خودکار

۶-۳. اجرای batchPrepare

```
conda activate gecoris
```

```
cd ~/sw/gecoris
python gecoris/batchPrepare.py /data/CR_PROJECT/config/snap_DSC124.parms
```

طبق مستند رسمی، Wrapper مربوطه SNAP را فراخوانی می‌کند تا Burstهای متناظر با AOI استخراج شوند و Precise Orbitها دریافت/اعمال شوند. خروجی Burstهای استخراج‌شده زیر workDir در پوشه slaves قرار می‌گیرد.

۶-۴. چه چیزهایی را نباید دستی انجام دهید؟

- قبل از SLC، GECORIS را Terrain Correction نکنید.
- آن را به GRD تبدیل نکنید.
- Complex data را با Exportهای تصویری از بین نبرید.
- اگر batchPrepare درست کار می‌کند، TOPSAR Split را دوباره دستی روی خروجی انجام ندهید.

۶-۵. کنترل خروجی slaves

- تعداد Acquisitionهای خروجی را با تعداد SLC ورودی مقایسه کنید.
- تاریخ هر محصول باید در محدوده startDate/endDate باشد.
- هر خروجی را در SNAP باز کنید و بررسی کنید AOI شما داخل آن قرار دارد.
- اگر یک تاریخ از قلم افتاده، Log همان تاریخ را جداگانه بررسی کنید؛ Pipeline را کورکورانه ادامه ندهید.

Checkpoint 3 — برای مسیر RCS/SCR/ALE، اگر slaves/ درست ساخته شده باشد می‌توانید بدون Co-registration وارد فصل Station Log و Monitoring شوید.

فصل ۷ - Master و Co-registration برای SCR Prediction و InSAR

۷-۱. چه زمانی Coreg لازم است؟

دو کاربرد به Stack هم‌مرجع نیاز دارند: SCR Prediction و CR-InSAR. مستندات صراحتاً می‌گویند Operational Monitoring بازتابندگی و Positioning Accuracy روی slaves اجرا می‌شود و Coregistration برای آن ضروری نیست.

۷-۲. انتخاب Master

```
python gecoris/getMaster.py /data/CR_PROJECT/config/snap_DSC124.parms
```

این مرحله Master را با هدف کاهش Baseline زمانی/هندسی انتخاب می‌کند. اگر masterDate را دستی تعیین کرده‌اید، مطمئن شوید آن Acquisition واقعاً در Dataset وجود دارد و کیفیت آن مناسب است.

۷-۳. اجرای Co-registration

```
python gecoris/batchCoreg.py /data/CR_PROJECT/config/snap_DSC124.parms
```

پیاده‌سازی مستندشده GECORIS از Co-registration، Single-Master مبتنی بر هندسه و ESD استفاده می‌کند و پردازش Subswath-wise است. از دید SNAP، منطق این مرحله معادل زنجیره‌ای از Back-Geocoding و Enhanced Spectral Diversity برای TOPS است.

۷-۴. کنترل Coregistration

- Master و Slave باید یک Pixel geometry مشترک داشته باشند.
- یک Scatterer قوی ثابت را روی چند تاریخ بررسی کنید؛ Peak نباید بین تصاویر جابه‌جایی پیکسلی بزرگ داشته باشد.
- اگر ESD شکست خورد، overlap Burstها، Subswath، Orbit و کیفیت Acquisition را بررسی کنید.
- Coregistration ضعیف، کل CR-InSAR را خراب می‌کند حتی اگر RCS ظاهراً خوب باشد.

فصل ۸ - تعریف Corner Reflector: reflectors.json و stacks.json

۸-۱. قالب رسمی reflectors.json

قالب سازگار با نمونه رسمی - مقادیر **CRS/FRAME** را از مثال اروپایی کپی نکنید

```
{
  "stations": [
    {
      "id": "CR01",
      "type": "CREF",
      "installDate": "20260410T1200Z",
      "startDate": "20260411T0000Z",
      "endDate": "99999999T9999Z",
      "ascending": {
        "orientation": true,
        "band": "*",
        "coordinates": {
          "longitude": 50.000000000,
          "latitude": 36.250000000,
          "elevation": 1350.000,
          "CRS": "YOUR_CRS",
          "FRAME": "YOUR_TRF",
          "EPOCH": "YOUR_EPOCH",
          "EPSG": "YOUR_EPSG"
        },
        "RCS0": null
      },
      "descending": {
        "orientation": true,
        "band": "*",
        "coordinates": {
          "longitude": 50.000000000,
          "latitude": 36.250000000,
          "elevation": 1350.000,
          "CRS": "YOUR_CRS",
          "FRAME": "YOUR_TRF",
          "EPOCH": "YOUR_EPOCH",
          "EPSG": "YOUR_EPSG"
        },
        "RCS0": null
```

```
      },
      "geometry": {
        "shape": "squareflipped",
        "flipped": true,
        "leglength": 0.76,
        "azimuthDip": 0.0,
        "zenithDip": 0.0
      },
      "RCS0": null
    }
  ]
}
```

برای پروژه در ایران — نمونه رسمی GECORIS از ETRS89/ETRF2000 استفاده می‌کند چون Case Study اروپایی است. این Frame را برای مختصات ایران کپی نکنید. برای ALE دقیق، مختصات Phase Center باید در یک Terrestrial Reference Frame مناسب با Epoch مشخص باشد و در صورت نیاز به Frame/epoch مدار Sentinel-1 تبدیل شود.

۸-۲. مختصات کدام نقطه **CR**؟

برای تحلیل ژئودتیکی دقیق، مختصات باید مربوط به Effective Phase Center یا Reference Point تعریف‌شده در طراحی CR باشد، نه یک گوشه فیزیکی دلخواه یا مختصات تقریبی پایه. اگر GNSS روی Monument جدا نصب می‌شود، Vector هندسی بین GNSS Reference Point و Phase Center باید معلوم باشد.

۸-۳. **startDate**، **installDate** و **endDate**

installDate زمان نصب فیزیکی است. startDate و endDate بازه‌ای را مشخص می‌کنند که Reflector در تحلیل Active در نظر گرفته می‌شود. تاریخ‌ها را با زمان UTC و قالب نمونه مستندات ثبت کنید. برای CRهایی که هنوز فعال‌اند مقدار انتهایی بزرگ مانند Template رسمی قابل استفاده است.

۸-۴. **Geometry**

shape، flipped، leglength و زاویه‌های Orientation باید مطابق هندسه واقعی سازه باشند. اگر شکل را اشتباه وارد کنید، RCS0 نظری و تحلیل تطبیقی بی‌معنا می‌شود. مقدار 0.76 متر و shape=squareflipped فقط مثال شبکه مقاله است، نه مقدار عمومی برای همه CRها.

۸-۵. **stacks.json**

```
{
  "stacks": [
    {
      "id": "s1_dsc124",
      "dataPath": "/data/CR_PROJECT/tracks/DSC124/",
      "sensor": "Sentinel-1",
      "subswath": "IW1",
      "type": "raw"
    },
    {
      "id": "s1_asc175",
      "dataPath": "/data/CR_PROJECT/tracks/ASC175/",
      "sensor": "Sentinel-1",
```

```
      "subswath": "IW2",
      "type": "raw"
    }
  ]
}
```

برای Monitoring عملیاتی روی slaves، type در Template رسمی raw است. برای کدها/شاخه‌های اصلاح‌شده‌ای که Stack Coregistered را مستقیم می‌خوانند ممکن است type=coreg استفاده شود؛ این بخش را با نسخه Repo خود تطبیق دهید و از حدس زدن Path خودداری کنید.

Checkpoint 4 — قبل از Monitoring، برای هر Station سه چیز را دستی تأیید کنید: مختصات، تاریخ فعال بودن، هندسه/ابعاد. برای هر Stack نیز subswath، dataPath و ID را تأیید کنید.

فصل ۹ - SCR Prediction: انتخاب محل قبل از نصب CR

۹-۱. منطق

SCR Prediction از Stack Coregistered موجود در دوره قبل از نصب استفاده می‌کند تا Clutter محلی را در یک Crop اطراف مختصات پیشنهادی برآورد کند و بر اساس RCS مورد انتظار CR، نقشه SCR پیش‌بینی‌شده بسازد. خروجی GeoTIFF است و می‌توان آن را در QGIS بررسی کرد.

۹-۲. planning.json

```
{
  "id": "SITE01",
  "longitude": 50.000000,
  "latitude": 36.250000,
  "elevation": 1350.0,
  "cropSize": 200,
  "RCS": 30,
  "stackDir": "/data/CR_PROJECT/tracks/DSC124/",
  "oversamplingFactor": 16,
  "outTiff": "/data/CR_PROJECT/output/scr_prediction/SCR_DSC124.tiff"
}
```

۹-۳. اجرای شبیه‌سازی

```
python gecoris/scrSimul.py /data/CR_PROJECT/config/planning_DSC124.json
```

۹-۴. تفسیر نقشه

مستندات عدد ثابت جهانی برای «قبول/رد» سایت تعیین نمی‌کنند؛ زیرا دقت مورد نیاز پروژه، هندسه CR و Clutter محیط متفاوت است. به‌جای یک Threshold دلخواه، چند Site را با RCS ثابت مقایسه کنید و نقاطی را ترجیح دهید که SCR

پیش‌بینی‌شده بالاتر و از Scatterer‌های قوی مجاور فاصله بیشتری دارند. سپس Accessibility، پایداری Monument، میدان دید، ایمنی و امکان جهت‌دهی CR را نیز وارد تصمیم کنید.

- نقشه را برای هر Track مورد استفاده جداگانه تولید کنید؛ یک محل ممکن است برای Descending خوب و برای Ascending بد باشد.
- اگر CR دوطرفه است، باید هر دو هندسه مشاهده بررسی شوند.
- نقشه SCR جایگزین بازدید میدانی نیست؛ فقط دید راداری را ارزیابی می‌کند.

فصل ۱۰ - Reflector Monitoring: استخراج SCR، RCS و ALE

۱۰-۱. gecoris.parms

```
{
  "project" : "CR_PROJECT",
  "stationLog" : "/data/CR_PROJECT/config/reflectors.json",
  "stackLog" : "/data/CR_PROJECT/config/stacks.json",
  "outDir" : "/data/CR_PROJECT/output/monitoring/",
  "precisePosFlag" : 1,
  "plotFlag" : 2,
  "atmoFlag" : 0,
  "ovsFactor" : 32
}
```

پارامتر	کارکرد
`precisePosFlag`	0 برای Positioning تقریبی؛ 1 برای Positioning دقیق
`plotFlag`	0 بدون Plot، 1 محدود، 2 جزئیات بیشتر
`atmoFlag`	فعال/غیرفعال کردن مدل خارجی تصحیح تأخیر جوی در Positioning
`ovsFactor`	FFT Oversampling برای تعیین Peak زیرپیکسلی؛ مثال رسمی 32

۱۰-۲. اجرای ماژول Monitoring

صفحه Reflector Monitoring مستندات، ساخت پارامتر و توابع پس از پردازش را نشان می‌دهد اما در نسخه مستندات عمومی نام Command اصلی Monitoring را به صراحت در همان صفحه چاپ نکرده است. بنابراین پس از Clone، نام Entry Script را در Root/پوشه gecoris نسخه خود کنترل کنید و فایل پارامتر را به همان Main Script بدهید. در شاخه‌های مشتق‌شده TU Delft، منطق تحلیل با پارامترهای stationLog/stackLog/outDir/precisePosFlag/ovsFactor اجرا می‌شود. از اجرای یک اسم Script حدسی روی نسخه متفاوت خودداری کنید.

چرا این احتیاط؟ — کد پژوهشی GECORIS و شاخه‌های بعدی آن طی زمان تغییر کرده‌اند. batchPrepare.py،

Entry در مستندات صراحتاً نام برده شده‌اند، اما getMaster.py، batchCoreg.py، scrSimul.py و CR_insar.py عمومی Monitoring Point در صفحه راهنما به همان وضوح ذکر نشده است.

۱۰-۳. منطق داخلی Monitoring

1. Station Log خوانده می‌شود و Reflectorها ساخته می‌شوند.
2. Stackهای SLC برای هر Orbit Track بارگذاری می‌شوند.
3. مختصات ژئودتیکی CR به Radar Datum هر Acquisition تبدیل می‌شود.
4. یک Crop اطراف محل پیش‌بینی‌شده استخراج می‌شود.
5. با FFT Oversampling محل Peak زیرپیکسلی و شدت پاسخ محاسبه می‌شود.
6. Outlierهای زمانی شناسایی می‌شوند.
7. RCS و Clutter/SCR برای هر Stack برآورد می‌شود.
8. در حالت Positioning دقیق، تصحیحات و ALE در Azimuth/Range محاسبه می‌شود.
9. Plotها و JSONهای ایستگاه و Stats صادر می‌شوند.

۱۰-۴. خروجی‌های مورد انتظار

استفاده	خروجی
داده پردازش‌شده ایستگاه/Time Series	`<station>.json`
خلاصه آماری RCS، SCR، ALE و عدم‌قطعیت	`<station>_stats.json`
کنترل سلامت زمانی Reflector	`RCS plot`
کنترل خطای موقعیت در Range/Azimuth	`ALE plot/TS`
مقایسه چند Reflector در شبکه	`Network plots`

فصل ۱۱ - تفسیر RCS، SCR و تشخیص مشکل

۱۱-۱. فیلدهای Stats

برداشت عملی	فیلد رسمی
RCS برآوردشده Reflector	`reflRCS`
`Signal-to-Clutter Ratio`	`SCR`
عدم‌قطعیت/پراکندگی برآورد RCS	`sigRCS`
Clutter پیش از نصب	`clutRCSbefore`

clutRCSafter	Clutter پس از نصب
sigPhi_DA	شاخص عدم‌قطعیت فاز از تحلیل زمانی
sigPhi_SCR	عدم‌قطعیت فاز وابسته به SCR
azimuthCRB	حد نظری دقت Azimuth
rangeCRB	حد نظری دقت Range
azimuthSTD / rangeSTD	پراکندگی زمانی موقعیت
azimuthALE / rangeALE	Bias/خطای مطلق مکان‌یابی
RCS0	RCS نظری/مرجع هندسی

۱۱-۲. چگونه **RCS** را قضاوت کنیم؟

به یک عدد منفرد نگاه نکنید. مهم‌ترین مقایسه‌ها عبارت‌اند از: RCS مشاهده‌شده نسبت به RCS0، ثبات RCS در زمان، تغییر ناگهانی پس از یک تاریخ و تفاوت بین Trackها. در Case Study مقاله، DBSTهای 0.76 متری با RCS نظری حدود 33.5 dBm^2 و RCS مشاهده‌شده نزدیک 33 dBm^2 گزارش شده‌اند؛ این فقط مثال همان طراحی و هندسه است و Threshold عمومی نیست.

۱۱-۳. علت افت ناگهانی **RCS**

- تغییر Orientation یا نشست/چرخش سازه CR.
- برف، آب، خاک، برگ یا debris داخل صفحات.
- خوردگی، تغییر شکل، پیچ یا اتصال شل.
- مانع جدید، خودرو، فنس یا سازه فلزی نزدیک.
- انتخاب Burst/Subswath/Track اشتباه یا تغییر Dataset.
- مختصات اشتباه که Crop را از Peak واقعی دور کرده است.

۱۱-۴. **SCR** و پروژه واقعی

SCR را به‌صورت نسبی و در ارتباط با دقت مورد نیاز تفسیر کنید. مقاله GECORIS نشان می‌دهد SCR مستقیماً با واریانس فاز و دقت Peak مرتبط است. در نمونه‌های مقاله، SCRهای حدود 22 تا 25 dB برای تعدادی Reflector گزارش شده‌اند. این اعداد Benchmark موردی هستند، نه قانون پذیرش همگانی.

فصل ۱۲ - Positioning دقیق و ALE: جایی که ژئودزی اهمیت پیدا می‌کند

۱۲-۱. چرا مختصات GNSS ساده کافی نیست؟

در Positioning ژئودتیکی دقیق، مختصات CR باید با Frame و Epoch مدار ماهواره سازگار باشد. مقاله GECORIS نشان می‌دهد در Case Study اروپا، بی‌توجهی به اختلاف ETRS89/ITRF در Epoch Acquisition می‌توانست خطای بیش از نیم متر ایجاد کند. برای منطقه شما نیز باید تحول Frame/Epoch متناسب با شبکه GNSS و ITRF مورد استفاده انجام شود.

۱۲-۲. تصحیحات کلیدی

- تبدیل مختصات Ground Target به TRF/epoch مناسب Acquisition.
- Solid Earth Tide برای موقعیت آنی هدف.
- Tropospheric slant delay؛ در کد از مدل‌های جوی خارجی مانند ERA5 استفاده می‌شود.
- Ionospheric delay بر اساس محصولات TEC/IONEX در Positioning دقیق.
- Sentinel-1 processor timing corrections شامل Doppler range، residual bistatic azimuth و FM-rate mismatch.

۱۲-۳. atmoFlag

اگر هدف شما فقط پایش سلامت CR است، atmoFlag=0 می‌تواند برای شروع کافی باشد. اگر هدف Calibration/ALE سانتی‌متری و تحلیل پژوهشی است، مدل جوی و تصحیحات دقیق اهمیت دارند. پیش از فعال کردن atmoFlag، دسترسی APIهای مربوط و Credentialها را در محیط قدیمی آزمایش کنید، چون سرویس‌های خارجی نسبت به ۲۰۲۱ تغییر کرده‌اند.

۱۲-۴. الگوی عیب‌یابی Bias

بررسی‌های اول	نشانه
Doppler/timing corrections، ارتفاع، جوی، Frame/Epoch	Bias ثابت چند متر در Range
زمان‌بندی، bistatic/FM-rate، مختصات/Frame	Bias ثابت در Azimuth
SCR پایین، peak contamination، orientation، clutter	پراکندگی بالا ولی Mean کم
SLC همان تاریخ، orbit، debris، weather/site change	Outlier فقط یک تاریخ

فصل ۱۳ - CR-InSAR: از Co-registered SLC تا تغییرمکان LOS

۱۳-۱. محدودیت مهم

مستندات GECORIS 0.1 تصریح می‌کنند که بخش InSAR در نسخه اصلی Research Code محدودیت دسترسی داشته است و توصیه می‌شود ابتدا InSAR AOI به‌صورت دستی و کنترل‌شده انجام شود. بنابراین CR-InSAR را به‌عنوان مرحله پیشرفته بعد از موفقیت Monitoring در نظر بگیرید، نه اولین تست نرم‌افزار.

۱۳-۲. فرمان خودکار مستند

```
python gecoris/CR_insar.py /data/CR_PROJECT/config/gecoris.parms
```

این Routine پارامترهای لازم را از فایل‌های پروژه می‌گیرد و برای Stackهای موجود پردازش را انجام می‌دهد. مستندات توصیه می‌کنند در Iteration دوم پارامترهای insar.parms متناسب با AOI تنظیم شوند.

۱۳-۳. مراحل مفهومی InSAR در GECORIS

1. استخراج Stack Coregistered به HDF5.
2. انتخاب Persistent Scatterer Candidates با Amplitude Dispersion/NAD.
3. ساخت Network و Arcها بین نقاط.
4. حل Temporal Ambiguities و حذف Outlier Arcها.
5. تعریف Reference Point/Datum.
6. برآورد و حذف Atmospheric Phase Screen در AOIهای مناسب.
7. حل دقیق شبکه و حذف Reference Phase Noise.
8. Densification با PSهای مرتبه دوم در صورت نیاز.
9. Refined geocoding و Export به CSV.
10. اتصال سری زمانی InSAR به Reflector و استخراج LOS displacement.

۱۳-۴. فایل‌های HDF5

در Workflow رسمی prepare_insar_HDF، فایل‌های InSAR stack_<stackId>.hdf5 و insar_<stackId>.hdf5 و JSON متادیتا را تولید می‌کند. HDF5 امکان نگهداری Candidate، SLC Stackها، شبکه و نتایج میانی را فراهم می‌کند. قبل از باز کردن Dataset جدید در Python، Dataset قبلی را close کنید تا Lock/Memory مشکل ایجاد نکند.

۱۳-۵. Reference Point

GECORIS سه منطق Reference را توضیح می‌دهد: auto، constraint و free. اگر یک CR با GNSS مطمئن دارید، constraint می‌تواند Reference را نزدیک مختصات آن انتخاب کند. انتخاب Reference یک تصمیم ژئودتیکی است؛ Time Series InSAR ذاتاً نسبی است و تفسیر مطلق بدون Datum مناسب اشتباه است.

۱۳-۶. LOS و تجزیه چند Track

خروجی اصلی هر Track تغییرمکان در Line-of-Sight است. اگر چند هندسه مشاهده مستقل در دسترس باشد، GECORIS مثال تجزیه LOS به مولفه Vertical و East-West را ارائه می‌کند. این تجزیه فقط زمانی معنی‌دار است که Geometry و زمان‌بندی Trackها و مدل حرکت برای پروژه مناسب باشد.

Checkpoint 5 — تا وقتی RCS/SCR پایدار و Coregistration قابل اعتماد نشده است، تغییرمکان InSAR CR را به‌عنوان نتیجه مهندسی گزارش نکنید.

فصل ۱۴ - کنترل کیفیت نهایی و چک‌لیست تحویل پروژه

۱۴-۱. کنترل داده

- تمام SLCها Product Type صحیح، Orbit و Relative Orbit یکسان در هر Stack دارند.
- هیچ Acquisition خراب یا ناقص در slaves/coreg وجود ندارد.
- Burst و Subswath واقعی CR را پوشش می‌دهد.
- تاریخ‌های قبل/بعد نصب درست برچسب‌گذاری شده‌اند.
- مختصات، ارتفاع، Frame و Epoch مستند و قابل ردیابی‌اند.
- Geometry و leglength از نقشه ساخت/اندازه‌گیری واقعی وارد شده‌اند.
- RCS Time Series و Plot Peak برای چند تاریخ به‌صورت بصری کنترل شده است.
- Outlierها دلیل‌یابی شده‌اند، نه اینکه صرفاً حذف شوند.
- برای ALE دقیق، تصحیحات مرجع و جوی مستند شده‌اند.
- برای InSAR، Reference Datum و Sign Convention LOS در گزارش نوشته شده است.

۱۴-۲. خروجی پیشنهادی گزارش مهندسی

بخش گزارش	محتوا
Data inventory	Track، dates، mode، polarization، product IDs
CR metadata	ID، نوع، ابعاد، Orientation، Phase Center coordinates
Processing	نسخه SNAP/Python/GECORIS، پارامترها، oversampling

Radiometry	RCS0، RCS observed، SCR، outliers
Geometry	ALE Range/Azimuth، frame/epoch، corrections
InSAR	reference، LOS TS، uncertainty، excluded epochs
QA	plots، عکس میدانی، log خطاها و تصمیم‌ها

فصل ۱۵ - خطاهای رایج و عیب‌یابی مرحله‌ای

۱۵-۱. **snappy import** نمی‌شود

- Python Environment اشتباه فعال شده است: which python را کنترل کنید.
- snappy-conf با Python دیگری اجرا شده است.
- نسخه SNAP و Python سازگار نیستند.
- jpy/Java path ناقص است.

۱۵-۲. **batchPrepare** چیزی در **slaves** نمی‌سازد

- AOI خارج از Footprint محصول است.
- swaths اشتباه است.
- dataDirs Path یا Tuple اشتباه نوشته شده است.
- startDate/endDate همه SLCها را فیلتر کرده‌اند.
- SNAP gpt یا Module Sentinel-1 در دسترس نیست.

۱۵-۳. **CR** در **Crop** دیده نمی‌شود

- مختصات Lat/Lon یا ارتفاع اشتباه.
- Reflector در تاریخ Acquisition هنوز نصب/فعال نبوده است.
- CR برای Orbit Direction دیگری جهت‌دهی شده است.
- Subswath/Burst نادرست انتخاب شده است.
- Orientation فیزیکی CR غلط یا پاسخ آن به علت Clutter پنهان شده است.

۱۵-۴. **Peak** چندگانه یا **SCR** ضعیف

- هدف فلزی/سازه قوی مجاور وجود دارد.
- CR نزدیک ساختمان، فنس یا دکل نصب شده است.
- Ground reflection/Multipath روی هندسه اثر دارد.

- Site planning باید دوباره با SCR map ارزیابی شود.

۱۵-۵. Coregistration/ESD شکست می‌خورد

- Master/Slave در Burst مشترک همپوشانی کافی ندارند.
- محصولی از Relative Orbit متفاوت وارد شده است.
- Subswath mismatch وجود دارد.
- یک Acquisition metadata/orbit مشکل دارد؛ آن را جداگانه تست کنید.

۱۵-۶. ALE چند متر است ولی Peak واضح است

این حالت معمولاً می‌گوید Radiometry خوب است اما مدل Positioning/Reference مشکل دارد. Frame/Epoch، ارتفاع Ellipsoidal، Phase Center offset، atmospheric delay، processor timing corrections و Time System را به ترتیب بررسی کنید.

۱۵-۷. پردازش روی S1C خطا می‌دهد

چون کد اصلی GECORIS قبل از Sentinel-1C نوشته شده، ابتدا همان Scene/Track را با S1A تست کنید. اگر S1A کار می‌کند و S1C نه، مسئله احتمالاً سازگاری parser/metadata است؛ به‌جای تغییر تصادفی داده، نسخه کد و Reader را بررسی کنید.

۱۵-۸. راه جایگزین مدرن برای Radar Coding

TU Delft در سال‌های بعد Toolbox دیگری به نام Delft Radar Coding Toolbox/Caroline Radar-Coding Toolbox منتشر کرده که صراحتاً خود را مبتنی بر GECORIS معرفی می‌کند و Installation جدیدتر مبتنی بر pixi دارد. این ابزار جایگزین کامل CR-InSAR نیست، اما اگر مشکل شما بخش Radar Coding/ALE و نصب Python/GDAL قدیمی است، بررسی آن می‌تواند مفید باشد. آن را با GECORIS اصلی مخلوط نکنید؛ یک Pipeline جدا و Version-controlled بسازید.

پیوست A - فایل‌های نمونه کامل برای شروع

A-1. snap_DSC124.parms

```
{
"SNAP_parameters" : {
    "snapPath" : "/home/user/snap/bin/",
    "repoPath" : "/home/user/sw/gecoris/gecoris/",
    "ram" : 24000,
    "cores" : 8
},
"Stack_parameters" : {
    "workDir" : "/data/CR_PROJECT/tracks/DSC124/",
    "dataDirs" : ("/data/CR_PROJECT/raw_s1/DSC124/",),
    "swaths" : ("IW1",),
    "min_lon" : "49.90",
    "max_lon" : "50.10",
    "min_lat" : "36.15",
    "max_lat" : "36.35",
    "startDate" : "20260101",
    "endDate" : "",
    "masterDate" : ""
}}
```

A-2. reflectors.json

```
{
"stations": [
{
  "id": "CR01",
  "type": "CREF",
  "installDate": "20260410T1200Z",
  "startDate": "20260411T0000Z",
  "endDate": "99999999T9999Z",
  "ascending": {
    "orientation": true,
    "band" : "*",
    "coordinates": {
      "longitude": 50.000000000,
      "latitude": 36.250000000,
      "elevation": 1350.000,
      "CRS": "YOUR_CRS",
      "FRAME": "YOUR_TRF",
      "EPOCH": "YOUR_EPOCH",
      "EPSG": "YOUR_EPSG"
    },
    "RCS0": null
  },
  "descending": {
    "orientation": true,
    "band" : "*",
    "coordinates": {
      "longitude": 50.000000000,
      "latitude": 36.250000000,
      "elevation": 1350.000,
      "CRS": "YOUR_CRS",
      "FRAME": "YOUR_TRF",
      "EPOCH": "YOUR_EPOCH",
      "EPSG": "YOUR_EPSG"
    },
    "RCS0": null
  },
  "geometry": {
    "shape": "squareflipped",
    "flipped": true,
    "leglength": 0.76,
    "azimuthDip": 0.0,
    "zenithDip": 0.0
```

```
  },
  "RCS0": null
}
]
}
```

A-3. stacks.json

```
{
"stacks": [
{
  "id": "s1_dsc124",
  "dataPath": "/data/CR_PROJECT/tracks/DSC124/",
  "sensor": "Sentinel-1",
  "subswath": "IW1",
  "type": "raw"
}
]
}
```

A-4. gecoris.parms

```
{
"project" : "CR_PROJECT",
"stationLog" : "/data/CR_PROJECT/config/reflectors.json",
"stackLog" : "/data/CR_PROJECT/config/stacks.json",
"outDir" : "/data/CR_PROJECT/output/monitoring/",
"precisePosFlag" : 1,
"plotFlag" : 2,
"atmoFlag" : 0,
"ovsFactor" : 32
}
```

A-5. planning.json

```
{
  "id": "SITE01",
  "longitude": 50.000000,
  "latitude": 36.250000,
  "elevation": 1350.0,
  "cropSize": 200,
  "RCS": 30,
  "stackDir": "/data/CR_PROJECT/tracks/DSC124/",
  "oversamplingFactor": 16,
  "outTiff": "/data/CR_PROJECT/output/scr_prediction/SCR_DSC124.tiff"
}
```

پیوست B - دستورات سریع از صفر تا خروجی

```
# 1) Environment
conda activate gecoris

# 2) Prepare raw SLC bursts
python gecoris/batchPrepare.py /data/CR_PROJECT/config/snap_DSC124.parms

# 3) If SCR Prediction or InSAR is needed
python gecoris/getMaster.py /data/CR_PROJECT/config/snap_DSC124.parms
python gecoris/batchCoreg.py /data/CR_PROJECT/config/snap_DSC124.parms

# 4) Pre-install site SCR prediction
python gecoris/scrSimul.py /data/CR_PROJECT/config/planning_DSC124.json

# 5) Reflector monitoring
# Run the monitoring entry script that exists in your checked-out GECORIS version,
# using gecoris.parms. Confirm the script name in the repository before execution.

# 6) CR-InSAR (documented automatic routine)
python gecoris/CR_insar.py /data/CR_PROJECT/config/gecoris.parms
```

B-2. ترتیب کنترل بعد از هر فرمان

مرحله	اگر درست بود	اگر درست نبود
batchPrepare	slaves با همه تاریخ‌ها	Path/AOI/swath/log را اصلاح کنید
getMaster	master تعیین شده	Dataset و baseline/date را بررسی کنید
batchCoreg	Stack هم‌مرجع	ESD/overlap/orbit را بررسی کنید
scrSimul	GeoTIFF تولید شده	stackDir/coreg/RCS/crop را بررسی کنید
Monitoring	JSON + RCS/ALE plots	station/stack/coords/burst را بررسی کنید
CR_insar	HDF/CSV + TS	اول Coreg و Manual InSAR QA را اصلاح کنید

پیوست C - منابع و مستندات مرجع

GECORIS Documentation - Introduction .1
https://gecoris.readthedocs.io/en/latest/introduction.html

GECORIS Documentation - Installation .2
https://gecoris.readthedocs.io/en/latest/installation.html

GECORIS Documentation - SNAP Processing .3
https://gecoris.readthedocs.io/en/latest/SNAP_processing.html

GECORIS Documentation - SCR Prediction .4
https://gecoris.readthedocs.io/en/latest/scr_prediction.html

GECORIS Documentation - Reflector Monitoring .5
https://gecoris.readthedocs.io/en/latest/reflector_monitoring.html

GECORIS Documentation - CR InSAR .6
https://gecoris.readthedocs.io/en/latest/reflector_insar.html

Czikhardt, R.; van der Marel, H.; Papco, J. (2021), Remote Sensing 13(5), 926 .7
https://doi.org/10.3390/rs13050926

Copernicus Data Space - Sentinel-1 collection .8
https://dataspace.copernicus.eu/data-collections/copernicus-sentinel-missions/sentinel-1

Copernicus Data Space - Sentinel-1 SLC documentation .9
https://documentation.dataspace.copernicus.eu/Data/Sentinel1.html

Copernicus Data Space - SLC Bursts .10
https://documentation.dataspace.copernicus.eu/APIs/Sentinel-1%20SLC%20Burst.html

ESA STEP - SNAP Download / current release .11
/https://step.esa.int/main/download/snap-download

TU Delft - Delft Radar Coding Toolbox (modern GECORIS-derived radar coding) .12
https://github.com/TUDelftGeodesy/DelftRadarCodingToolbox

یادداشت نسخه

این راهنما بر اساس مستندات قابل دسترس در ۲۱ سپتامبر ۲۰۲۶ تدوین شده است. بخش‌های نصب GECORIS بر نسخه پژوهشی قدیمی تکیه دارند و سرویس‌های خارجی، APIها، SNAP و Python ممکن است تغییر کنند. برای پروژه تولیدی، نسخه دقیق نرم‌افزارها و Hash کد را در گزارش ثبت کنید و Environment را Freeze کنید.

جمع‌بندی اجرایی — برای اولین موفقیت، پروژه را کوچک کنید: یک Track، یک CR، چند SLC، atmoFlag=0 و Monitoring خام. بعد از اینکه Peak، RCS و SCR را درست دیدید، به Positioning، SCR map، Coregistration دقیق و در نهایت CR-InSAR بروید.