# Scheda di conformità — star-audit

Tutto ciò che una qualifica fornitore, una revisione legale o un controllo sulle esportazioni chiede di solito su star-audit 0.13.1, in un solo documento.

## 1. Identità

| | |
|---|---|
| Documento | STAR-PKG-STAR-AUDIT-IT · v2.3 · 2026-11-14 |
| Pacchetto | star-audit |
| Versione | 0.13.1 |
| Capacità | star.audit.geodesy |
| Maturità | Beta (VERIFIED) |
| Python | >=3.9 |
| Commit del sorgente | dc09f61d |
| Sorgente | https://github.com/Andrea07072000/S.T.A.R.-/tree/main/packages/star_audit |

## 2. Licenza e obblighi

Licenza Apache 2.0. Puoi usarlo, copiarlo, modificarlo e distribuirlo, anche a fini commerciali, e ricevi una licenza sui brevetti dei contributori. Se lo ridistribuisci: conserva il file LICENSE e le note sul diritto d'autore (e un file NOTICE, se una versione futura lo aggiunge), e indica nei file modificati che li hai cambiati. Non puoi usare il nome S.T.A.R. per promuovere il tuo prodotto.

## 3. Che cosa richiede per funzionare

| RICHIEDE | LICENZA | FONTE |
|---|---|---|
| star-geodesy | Apache-2.0 | S.T.A.R. |
| mpmath | BSD-3-Clause | https://mpmath.org/ |

# 4. Integrità

Impronte SHA-256 dei file pubblicati su questo sito. Per controllare un file scaricato:

```
python -c "import
hashlib,sys;print(hashlib.sha256(open(sys.argv[1],'rb').read()).hexdigest())" star_audit-
0.13.1-py3-none-any.whl
```

| FILE | SHA-256 |
|---|---|
| star_audit-0.13.1-py3-none-any.whl | a6740807909318783d61d5c1c5889622621ca9a439bb5841b55531ab8f75d978 |
| star_audit-0.13.1-source.zip | 1985fe4c3b1cff42481ff2078cfa02f7ac5fa0a0524bea6eea1c61246f2d529c |

# 5. Che cosa fa, e i suoi limiti

Misura l'inviluppo di accuratezza (per fascia di quota, 0-36.000 km) di qualunque libreria ECEF->geodetiche rispetto a un riferimento esatto.

Che queste librerie siano «sbagliate»: possono essere tarate per l'uso vicino alla superficie; altre opzioni di chiamata non sono state esaminate.

- `audit_function(fn)` / `run_external(python, code)`: 450 punti (10 latitudini x 5 longitudini x 9 quote da 0 a 36.000 km) con coordinate geodetiche NOTE; la libreria converte le loro coordinate ECEF esatte; gli errori sono misurati rispetto al valore vero.
- Inviluppo per fascia di quota: errore massimo in latitudine (gradi), in quota (m), in posizione 3-D (m); caso peggiore.
- La libreria gira in un proprio interprete (`run_external`); un'esecuzione della libreria che fallisce solleva un errore, non restituisce mai un punteggio.
- `audit(python, driver)`: usa una libreria di tempo SOLO attraverso la sua API documentata stringa-UTC -> stringa-TAI su 7 sonde (inclusi istanti dentro 23:59:60) e classifica OK / WRONG / REJECTED / PAST_TABLE_WARNED / PAST_TABLE_SILENT rispetto a un valore vero scritto a mano dal Bollettino C dell'IERS.
- `audit(command, driver, cases)` contro Vallado SGP4-VER (33 casi, 667 epoche; dati di prova con provenienza): per caso errore 3-D massimo, mediana, rifiuti. I casi sono appaiati per ORDINE (NORAD 20413 compare due volte: finestra breve e arco di 3,5 anni).
- `audit(impls, cases)`: ogni implementazione gira nel proprio interprete (driver `teme_to_gcrs(utc, r_km)`); prima la validazione della sonda: l'esempio TEME->GCRF pubblicato da Vallado (2004-04-06

07:51:28.386009 UTC) deve essere riprodotto entro 1 m; poi inviluppi di divergenza a coppie per anno 1980-2050 su vettori LEO e GEO, e rifiuti.

- `audit(impls, cases)`: corpus deterministico (seme 20261004, fissato con sha256) di 40 frame TM validi (CCSDS 132.0-B, ogni opzione d'intestazione: OCF, intestazione secondaria da 1 a 8 ottetti) e 260 copie con da 1 a 4 bit invertiti nell'intestazione, nei dati o nel FECF. Validazione della sonda: un'implementazione conta solo se decodifica ogni frame valido campo per campo; un'eccezione del driver non è mai un rifiuto credibile. Riporta l'accettazione silenziosa di frame con FECF errato, le collisioni del CRC e i disaccordi di accettazione.
- `audit(impls, epochs)`: ogni implementazione nel proprio interprete (`tdb_minus_tt(jd_tt)`); validazione della sonda: ogni epoca entro 50 us dalla formula pubblicata a bassa precisione (Explanatory Supplement, 0.001657 sin g + 0.000014 sin 2g); inviluppi a coppie su 401 epoche 1900-2100, per secolo; i rifiuti escludono.
- `audit(impls, epochs)`: ogni implementazione nel proprio interprete (`gmst_deg(jd_ut1)`); validazione della sonda sull'esempio 3-5 pubblicato da Vallado (GMST = 152,578787810 gradi al 1992-08-20 12:14:00 UT1) entro 0,1 secondi d'arco; inviluppi a coppie in secondi d'arco, sicuri rispetto al giro completo, su 201 epoche 1950-2050, per mezzo secolo; i rifiuti escludono.
- `audit(impls, cases)`: rv -> (p, e, i, RAAN, argp, nu) per implementazione nel proprio interprete; validazione della sonda sull'esempio 2-5 di Vallado (tolleranze 0,01 km / 1e-5 / 0,01 gradi, dimensionate sulle cifre stampate); corpus con seme fisso, fissato con sha256, di 60 stati regolari + 20 singolari (quasi circolari, quasi equatoriali, entrambi); i casi regolari sono confrontati con il valore vero generatore e a coppie; casi singolari: la CONVENZIONE di ciascuna implementazione (0, valore sentinella, NaN, valore, rifiuto).
- `audit(impls, cases)`: che cosa accettano i lettori di TLE. Riferimento = la regola del formato (checksum modulo 10 con '-' = 1, numeri di riga, numeri di satellite uguali, 69 colonne). Corpus: i 29 TLE ben formati di SGP4-VER più 6 tipi derivati (field_digit, checksum, line_number, satnum_mismatch, truncated = da rifiutare; resummed = ben formato, da accettare), fissato con sha256. Validazione della sonda: ogni TLE valido accettato, l'inclinazione letta entro 1e-4 gradi e il numero di catalogo letto esattamente (un'inclinazione NaN o un numero sbagliato sono una lettura errata: prima della 0.13.1 entrambi venivano validati).
- `audit(impls, cases)`: che cosa accettano i decodificatori di Space Packet (regola CCSDS 133.0-B: versione '000', campo lunghezza = ottetti di dati - 1). Corpus (fissato con lo sha256 del corpus arrotondato): 40 validi + 10 idle (da accettare) e 4 x 40 difettosi (versione, troncato, in coda, solo intestazione: da rifiutare). Validazione della sonda: ogni pacchetto valido decodificato campo per campo.
- `audit(impls, cases)`: risolutori di Keplero ellittici M -> E. Validazione della sonda: esempio 2-1 di Vallado (M = 235,4 gradi, e = 0,4 -> E = 220,512074767522 gradi, tolleranza 1e-9 rad). Valore vero: 8 eccentricità (da 0 a 0,9999) x 41 anomalie, l'inverso esatto (40 cifre, mpmath) del double M inviato a ciascuna libreria; errore circolare. Una risposta non finita su un caso regolare è un rifiuto. Gli ingressi ostili (NaN, inf, e = 1, e > 1, e < 0, M enorme o negativo) sono registrati come categoria di comportamento per libreria. Impronta del corpus fissata.

- `audit(impls, cases)`: risolutori di Lambert a zero rivoluzioni, driver `lambert(r1, r2, tof, mu) -> v1` (prograda). Validazione della sonda: esempio 5.2 di Curtis (v1 = (-5,9925, 1,9254, 3,2456) km/s, tolleranza 1e-4 km/s). Valore vero senza alcun risolutore di Lambert: 315 casi costruiti da orbite note (3 semiassi maggiori x 5 eccentricità fino a 0,95 x 3 inclinazioni x 7 angoli di trasferimento da 10 a 330 gradi); r1, r2 e il v1 vero dalle equazioni perifocali in forma chiusa, tempo di volo dalla catena diretta nu -> E -> M. Una risposta non finita o di lunghezza sbagliata è un rifiuto. Gli ingressi ostili (NaN, tof <= 0, vettore nullo, trasferimenti a 0 e 180 gradi) sono registrati per libreria. Impronta del corpus fissata.
- `audit(impls, cases)`: propagatori kepleriani, driver `propagate(r0, v0, tof, mu) -> [x, y, z, vx, vy, vz]`. Validazione della sonda: esempio 2-4 di Vallado (1e-3 km, 1e-5 km/s; la costante è stata confrontata con NAIF prop2b). Valore vero senza alcun propagatore: 450 orbite note (3 semiassi maggiori x 5 eccentricità fino a 0,95 x 3 inclinazioni x 5 angoli di trasferimento x 0 o 10 rivoluzioni intere), stati dalle equazioni perifocali in forma chiusa. Una risposta non finita o di lunghezza sbagliata è un rifiuto. Ogni ingresso ostile gira da solo con un limite di 120 s; l'assenza di risposta è registrata come `error:Hang`.
- Ogni auditor legge le righe stampate da un driver attraverso un'unica guardia: esattamente un risultato per caso, sull'ultima riga di uscita, altrimenti `RuntimeError`. Trovato da una sonda ostile (2026-10-06): un lettore che rispondeva a 1 caso su 203 veniva validato (i totali si contavano su ciò che tornava), e un'uscita vuota o non JSON sollevava eccezioni non dichiarate.

# 6. Esportazione, uso e certificazione

Il pacchetto è pubblicato e liberamente disponibile. Di solito questo esclude il software dagli elenchi di controllo delle esportazioni, ma non ogni suo uso: leggi la pagina su attività spaziali ed esportazioni prima di usarlo in un programma soggetto a controllo. Non è certificato né qualificato per il volo; le evidenze descrivono che cosa è stato provato e come.

# 7. Citazione e distinta dei componenti

```
@software{cavazzini_star_audit_0_13_1,
  author  = {Cavazzini, Andrea},
  title   = {{star-audit}: Audit dell'accuratezza delle conversioni geodetiche},
  version = {0.13.1},
  year    = {2026},
  url     = {https://andrea07072000.github.io/S.T.A.R.-/software/star_audit/},
  note    = {Apache-2.0. Source: https://github.com/Andrea07072000/S.T.A.R.-, commit dc09f61d}
}
```

CITATION.cff · BibTeX · SBOM CycloneDX 1.5 · SBOM SPDX 2.3

# 8. Garanzia e responsabilità

Fornito «così com'è», senza garanzie o condizioni di alcun tipo (licenza Apache 2.0, sezioni 7 e 8), nei limiti consentiti dalla legge. Ciò non esclude mai la responsabilità che la legge non consente di escludere.

Collegati: attività spaziali ed esportazioni · uso accettabile · evidenze e limiti · testi delle licenze