

Architettura Computazionale e Visualizzazione delle Mappe di Confidenza $\Delta\chi^2$ 2D

Manuale Tecnico e Guida alle API del Modulo `chi_squared_2d_map`

Laboratorio di Fisica Sperimentale e Computazionale

21 ottobre 2026

Sommario

Questo manuale fornisce la documentazione tecnica e operativa del modulo `chi_squared_2d_map.py`, progettato per l'esplorazione e la rappresentazione grafica rigorosa delle superfici di costo bi-dimensionali scaturite da processi di fit ai minimi quadrati generalizzati (ODR Metodo A). Vengono analizzati in dettaglio: l'algoritmo di profilazione vincolata con scansione a serpentina e *warm-start* sequenziale di `Minuit`, l'interpolazione spline cubica per la ricostruzione continua della varietà, la topologia della griglia asimmetrica a quattro sub-riquadri via `GridSpec`, l'isolamento del box statistico di covarianza nel quadrante orfano e l'integrazione con i profili marginali 1D calcolati dall'algoritmo MINOS.

Indice

1	Introduzione e Finalità del Modulo	2
2	Algoritmo di Profilazione e Ricostruzione della Superficie	2
2.1	Campionamento a Serpentina e Warm-Start Sequenziale	2
2.2	Regolarizzazione e Interpolazione Spline 2D	3
3	Topologia del Canvas e Geometria dei Subplot	4
3.1	Partizione Asimmetrica con <code>gridspec.GridSpec</code>	4
3.2	Sincronizzazione degli Assi e Soppressione Sovrapposizioni	4
4	Livelli di Confidenza, Contorni e Segnaletica	6
4.1	Significato Statistico dei Valori $\Delta\chi^2$	6
4.2	Verifica Geometrica delle Tangenti MINOS	6
4.3	Leggibilità del Testo con Effetto Contorno (<i>Halo Stroke</i>)	6
5	Firma Completa delle API: <code>chi2_map_pipeline</code>	7
5.1	Dizionario dei Parametri di Configurazione	7
6	Integrazione con la Pipeline di Fit ed Esempio Completo	8

1 Introduzione e Finalità del Modulo

Nelle analisi di fisica sperimentale, l'esame della sola matrice di covarianza $V = 2\mathbf{H}^{-1}$ (calcolata localmente nel minimo con `hesse`) fornisce una descrizione parabolica locale che ignora le asimmetrie e le correlazioni non lineari dello spazio dei parametri. Per visualizzare la reale geometria dell'incertezza congiunta di due parametri (θ_x, θ_y) , il modulo `chi_squared_2d_map.py` implementa:

1. **Scansione Profilata Reale (Profile Likelihood Scan):** a ogni nodo della griglia 2D i parametri θ_x e θ_y vengono fissati, mentre tutti i restanti $p - 2$ parametri liberi del modello vengono rilassati e ri-ottimizzati tramite `migrad()`.
2. **Sincronizzazione Geometrica 1D–2D:** proiezione sui bordi dei profili unidimensionali MINOS calcolati da `mnprofile`, verificando visivamente che le rette tangenti al contorno $1\sigma_{1D}$ ($\Delta\chi^2 = 1.0$) coincidano con gli errori asimmetrici marginali.
3. **Gestione Integrata dello Spazio Morto:** allocazione del box statistico con $\text{Cov}(\theta_x, \theta_y)$ e ρ_{xy} nel quadrante in alto a destra, eliminando interferenze con i dati.
4. **Respiro Laterale della Colorbar:** posizionamento di un canale dedicato a destra con cuscinetto (*buffer pad*), separando la barra di calibrazione continua dal profilo marginale verticale.

2 Algoritmo di Profilazione e Ricostruzione della Superficie

2.1 Campionamento a Serpentina e Warm-Start Sequenziale

Dato un modello a p parametri con minimo globale $F_{\min} = \chi^2(\hat{\boldsymbol{\theta}})$, si definisce un dominio di interesse bidimensionale centrato su $(\hat{\theta}_x, \hat{\theta}_y)$ di semi-ampiezza N_σ :

$$x \in [\hat{\theta}_x - N_\sigma\sigma_x, \hat{\theta}_x + N_\sigma\sigma_x], \quad y \in [\hat{\theta}_y - N_\sigma\sigma_y, \hat{\theta}_y + N_\sigma\sigma_y] \quad (1)$$

dove σ_x e σ_y rappresentano le incertezze massime (o MINOS asimmetriche estese) estratte dall'oggetto `iminuit.Minuit`.

Il dominio viene discretizzato in una griglia regolare $K \times K$ (tipicamente $K \in [30, 45]$). La minimizzazione ripetuta a ogni nodo richiede una gestione accurata del punto iniziale per prevenire salti su minimi locali o fallimenti dell'algoritmo di discesa. Il modulo adotta una traiettoria a **serpentina alternata** (*boustrophedon*):

```
1 for i in range(grid_size):
2     # Alterna la direzione di scansione per massimizzare la vicinanza metrica
3     j_indices = range(grid_size) if (i % 2 == 0) else reversed(range(grid_size))
4     for j in j_indices:
5         # 1. Warm-start: inizializza i p-2 parametri dallo stato ottimo del vicino
6         for p in param_names:
7             m.values[p] = current_cache[p]
8
9         # 2. Fissazione della coppia bidimensionale
10        m.fixed[param_x] = True
11        m.fixed[param_y] = True
12        m.values[param_x] = X_map[i, j]
13        m.values[param_y] = Y_map[i, j]
14        m.migrad()
15
16        # 3. Fallback di sicurezza se la discesa diverge o non e' valida
17        if not m.valid:
18            for idx, p in enumerate(param_names):
19                m.values[p] = orig_values[idx]
20                m.values[param_x] = X_map[i, j]
```

```

21         m.values[param_y] = Y_map[i, j]
22         m.migrad()
23
24     Z_profiled[i, j] = m.fval
25     for p in param_names:
26         current_cache[p] = m.values[p]

```

Listing 1: Scansione a serpentina con warm-start e fallback di ripristino.

Il *warm-start* riduce di oltre il 70% le iterazioni interne di `migrad`, poiché il vettore dei parametri ottimi a coordinate (i, j) costituisce un’eccellente approssimazione di partenza per il nodo adiacente $(i, j \pm 1)$.

2.2 Regularizzazione e Interpolazione Spline 2D

La matrice campionata discretamente $Z_{ij} = \chi^2(x_i, y_j)$ viene riferita al minimo assoluto:

$$\Delta Z_{ij} = Z_{ij} - F_{\min} \quad (2)$$

Per tracciare curve di livello continue ed evitare seghettature poligonali nei contorni chiusi, la matrice ΔZ viene sovracampionata mediante interpolazione spline cubica (`scipy.ndimage.zoom`) con fattore di moltiplicazione Z_f (default: 4):

```

1 dz_smooth = ndimage.zoom(dz, zoom_factor, order=3)

```

La matrice continua risultante possiede dimensione $(K \cdot Z_f) \times (K \cdot Z_f)$, garantendo contorni ellittici fluidi e privi di artefatti numerici di discretizzazione.

3 Topologia del Canvas e Geometria dei Subplot

3.1 Partizione Asimmetrica con gridspec.GridSpec

La disposizione simultanea della mappa 2D, dei due profili 1D e della barra colori richiede una suddivisione rigorosa dello spazio priva di automatismi coprenti. Con l'opzione standard `colorbar.loc="right"`, la figura viene ripartita in una griglia di 2 righe $\times$ 4 colonne:

```

1 fig = plt.figure(figsize=(8.2, 7.2))
2 gs = gridspec.GridSpec(
3     2, 4,
4     width_ratios=[3.5, 1.0, 0.1, 0.15],
5     height_ratios=[1.0, 3.5]
6 )
7 gs.update(wspace=spacing, hspace=spacing)
8
9 ax1 = fig.add_subplot(gs[1, 0])           # Mappa principale 2D
10 ax_t = fig.add_subplot(gs[0, 0], sharex=ax1) # Profilo 1D superiore
11 ax_r = fig.add_subplot(gs[1, 1], sharey=ax1) # Profilo 1D laterale destro
12 cax = fig.add_subplot(gs[1, 3])           # Canale Colorbar dedicato

```

Listing 2: Definizione della griglia proporzionale con colorbar a destra.

Subplot / Slot	Righe / Colonne	Ratio ($H \times W$)	Contenuto e Vincoli Fisici
ax_t	[0, 0]	1.0×3.5	Profilo $\Delta\chi^2$ parametro x ; asse x condiviso con ax1.
<i>Quadrante Orfano</i>	[0, 1]	1.0×1.0	Riquadro statistico: $\text{Cov}(\theta_x, \theta_y)$, ρ_{xy} , errori asimmetrici.
<i>Buffer Pad</i>	[0, 2] e [1, 2]	1.0×0.1	Intervallo vuoto di distacco per isolare la colorbar.
ax1	[1, 0]	3.5×3.5	Mappa di densità $\Delta\chi^2$, livelli di contorno e punto ottimo (+).
ax_r	[1, 1]	3.5×1.0	Profilo $\Delta\chi^2$ parametro y ; asse y condiviso con ax1.
cax	[1, 3]	3.5×0.15	Scala cromatica graduata continua con tacche sui livelli σ_{2D} .

Tabella 1: Mappatura delle celle della griglia GridSpec in configurazione laterale.

3.2 Sincronizzazione degli Assi e Soppressione Sovrapposizioni

La coerenza visiva esige l'azzeramento dello spazio tra la mappa e i profili (`wspace=0.01`, `hspace=0.01`):

- L'asse orizzontale di `ax_t` condivide i limiti con `ax1` (`sharex=ax1`), ma le etichette numeriche inferiori vengono soppresse:

```

1 ax_t.tick_params(axis='x', bottom=True, labelbottom=False)
2

```

- L'asse verticale di `ax_r` condivide i limiti con `ax1` (`sharey=ax1`), sopprimendo le etichette numeriche a sinistra:

```

1 ax_r.tick_params(axis='y', left=True, labelleft=False)
2

```

- **Soppressione del Tick Zero Sovrapposto:** per evitare che la cifra 0 sull'asse $\Delta\chi^2$ di `ax_r` si sovrapponga visivamente all'ultimo tick numerico dell'asse x di `ax1`, è possibile applicare un formattatore di potatura:

```
1 ax_r.xaxis.set_major_formatter(  
2     ticker.FuncFormatter(lambda v, _: " " if abs(v) < 1e-3 else f"{v:.0f}")  
3 )  
4
```

4 Livelli di Confidenza, Contorni e Segnaletica

4.1 Significato Statistico dei Valori $\Delta\chi^2$

Nella rappresentazione congiunta di due parametri, la variabile $\Delta\chi^2(\theta_x, \theta_y)$ segue asintoticamente una distribuzione chi-quadro a $\nu = 2$ gradi di libertà:

$$P(\Delta\chi^2 \leq \Delta\chi_0^2) = \int_0^{\Delta\chi_0^2} \frac{1}{2} e^{-z/2} dz = 1 - e^{-\Delta\chi_0^2/2} \quad (3)$$

Al contrario, per un profilo marginale monodimensionale (proiettato su un singolo asse rilassando l'altro), i gradi di libertà sono $\nu = 1$:

$$P(\Delta\chi^2 \leq \Delta\chi_0^2) = \text{erf} \left(\sqrt{\frac{\Delta\chi_0^2}{2}} \right) \quad (4)$$

Livello Tracciato	Gradi di Libertà	$\Delta\chi^2$	Copertura	Stile Grafico Adottato
$1\sigma_{1D}$ (Marginale)	$\nu = 1$	1.00	68.27%	Tratto-punto nero ('-.', spessore 1.1 pt)
$1\sigma_{2D}$ (Congiunto)	$\nu = 2$	2.30	68.27%	Continuo bianco ('-', spessore 1.6 pt)
$2\sigma_{2D}$ (Congiunto)	$\nu = 2$	6.18	95.45%	Tratteggiato bianco ('--', spessore 1.2 pt)
$3\sigma_{2D}$ (Congiunto)	$\nu = 2$	11.83	99.73%	Punteggiato bianco (':', spessore 0.9 pt)

Tabella 2: Livelli di confidenza standard e stili geometrici implementati nel modulo.

4.2 Verifica Geometrica delle Tangenti MINOS

Sul piano 2D, il contorno ad anello $1\sigma_{1D}$ ($\Delta\chi^2 = 1.00$) non racchiude il 68.27% della probabilità congiunta (che richiederebbe $\Delta\chi^2 = 2.30$), bensì possiede una proprietà geometrica fondamentale:

Le rette tangenti orizzontali e verticali al contorno $\Delta\chi^2 = 1.00$ individuano esattamente gli intervalli di incertezza marginale a 1σ calcolati dal solutore MINOS per i singoli parametri.

Il modulo traccia tali tangenti mediante linee rosse tratteggiate (`ls='--'`, `lw=1.2`), che si prolungano simultaneamente sia nella mappa centrale sia nei subplot 1D marginali, evidenziando visivamente la concordanza tra la scansione 2D e i profili parabolici/asimmetrici 1D.

4.3 Leggibilità del Testo con Effetto Contorno (*Halo Stroke*)

Per garantire che le etichette numeriche dei livelli di contorno (es. $1\sigma_{2D}, 2\sigma_{2D}$) rimangano perfettamente visibili sia su sfondi scuri che su regioni a saturazione luminosa (come accade con le mappe `viridis_r` o `plasma`), il testo viene corredato da un effetto contorno (*path stroke*):

```

1 halo = [patheffects.withStroke(linewidth=2.5, foreground="white")]
2 clbl_2d = ax1.clabel(cs_2d, inline=True, fmt=fmt_clabel, colors="black")
3 plt.setp(clbl_2d, path_effects=halo)
4
5 clbl_1d = ax1.clabel(cs_1d, inline=True, fmt=fmt_clabel, colors="black")
6 plt.setp(clbl_1d, path_effects=halo)

```

Listing 3: Applicazione del filtro tipografico halo con `patheffects`.

5 Firma Completa delle API: chi2_map_pipeline

```

1 def chi2_map_pipeline(
2     risultati: Any,
3     param_x: str,
4     param_y: str,
5     param_info: Optional[Dict[str, Dict[str, str]]] = None,
6     n_sigma: float = 3.5,
7     grid_size: int = 35,
8     zoom_factor: int = 4,
9     show_clabels: bool = True,
10    manual_clabels: Union[bool, List[Tuple[float, float]]] = False,
11    clabel_fontsize: float = typo_params["legendsize"],
12    stats_box_loc: str = "inside_map", # 'inside_map', 'top', o 'custom'
13    stats_box_pos: Optional[Tuple[float, float]] = None,
14    show_stats_box: bool = True,
15    spacing: float = 0.01,
16    cmap: str = "viridis_r",
17    save_path: Optional[str] = None,
18    show: bool = True,
19    colorbar_loc: str = "right", # 'right', 'bottom', oppure 'none'
20 ) -> Tuple[plt.Figure, Tuple[plt.Axes, plt.Axes, plt.Axes]]:

```

Listing 4: Firma della funzione principale chi2_map_pipeline.

5.1 Dizionario dei Parametri di Configurazione

Parametro	Tipo / Default	Descrizione e Funzione Operativa
risultati	RisultatiFit	Contenitore dell'output del fit contenente l'istanza minuit_obj.
param_x, param_y	str	Nomi delle due variabili fisiche da scansionare (es. "m", "q").
param_info	dict / None	Metadati di visualizzazione: simbolo LaTeX ('label') e unità di misura ('unit').
n_sigma	float = 3.5	Semi-estensione dell'intervallo di scansione espressa in deviazioni standard.
grid_size	int = 35	Risoluzione della griglia nodale $K \times K$ prima della regolarizzazione.
zoom_factor	int = 4	Fattore di interpolazione spline cubica ($K \cdot Z_f$ punti finali).
show_clabels	bool = True	Abilita la scrittura a video delle etichette dei contorni ($1\sigma_{2D}$, ecc.).
manual_clabels	bool / list	Coordinate manuali delle etichette per evitare sovrapposizioni su curve ripide.
stats_box_loc	str = "inside_map"	Posizione del box di correlazione: 'inside_map', 'top', 'custom'.
stats_box_pos	tuple / None	Coordinate (x, y) in frazione di figura per la modalità 'custom'.
colorbar_loc	str = "right"	Posizione della scala colori: 'right', 'bottom' o disabilitata.
spacing	float = 0.01	Separazione orizzontale e verticale tra i riquadri (wspace, hspace).
cmap	str = "viridis_r"	Mappa cromatica continua di Matplotlib per i valori del $\Delta\chi^2$.
save_path	str / Path / None	Percorso su disco per l'esportazione automatica in formato vettoriale PDF/SVG.

Tabella 3: Parametri di configurazione del generatore di mappe profilate.

6 Integrazione con la Pipeline di Fit ed Esempio Completo

Lo script seguente illustra il ciclo completo di laboratorio: generazione dati con errori bivariati, stima non distorta con FitPipeline (ODR Metodo A) e produzione della mappa con box statistico nel quadrante in alto a destra (`stats.box_loc="custom"`).

```
1 import numpy as np
2 from pathlib import Path
3 from chi_squared_fit import FitPipeline
4 from chi_squared_2d_map import chi2_map_pipeline
5
6 # 1. Definizione della funzione teorica (Legge di Ohm con offset)
7 def modello_lineare(x, m, q):
8     return m * x + q
9
10 # 2. Generazione del campione sintetico con errori su entrambe le variabili
11 rng = np.random.default_rng(123)
12 m_vero, q_vero = 2.45, 1.10
13 n_punti = 35
14
15 x_pulito = np.linspace(1.0, 10.0, n_punti)
16 y_pulito = modello_lineare(x_pulito, m_vero, q_vero)
17
18 err_x = np.full(n_punti, 0.20)
19 err_y = np.full(n_punti, 0.45)
20
21 x_misurato = x_pulito + rng.normal(0, err_x, n_punti)
22 y_misurato = y_pulito + rng.normal(0, err_y, n_punti)
23
24 # 3. Esecuzione FitPipeline con Metodo A (Varianza Congelata e MINOS)
25 pipeline = FitPipeline(
26     model=modello_lineare,
27     param_names=["m", "q"],
28     p0=[1.0, 0.0],
29     run_minos=True,
30     verbose=True,
31 )
32
33 risultati_fit = pipeline.fit(
34     x=x_misurato,
35     y=y_misurato,
36     err_x=err_x,
37     err_y=err_y,
38     odr_lin_Newton=True, # Attiva il Metodo A non distorto
39 )
40
41 # 4. Configurazione dei metadati dimensionali
42 info_parametri = {
43     "m": {"label": r"$m$", "unit": r"\mathrm{V/s}"},
44     "q": {"label": r"$q$", "unit": r"\mathrm{V}"},
45 }
46
47 # 5. Generazione della Mappa 2D ad alta risoluzione
48 save_folder = Path("examples")
49 save_folder.mkdir(parents=True, exist_ok=True)
50
51 fig, (ax_main, ax_top, ax_right) = chi2_map_pipeline(
52     risultati_fit,
53     param_x="m",
54     param_y="q",
55     param_info=info_parametri,
56     n_sigma=3.2,
57     grid_size=32,
```

```

58     zoom_factor=4,
59     stats_box_loc="custom",
60     stats_box_pos=(0.725, 0.85), # Quadrante in alto a destra
61     show_clabels=True,
62     manual_clabels=False,
63     colorbar_loc="right",
64     cmap="viridis_r",
65     spacing=0.0,
66     save_path=save_folder / "mappa_chi2_profilata.pdf",
67     show=True,
68 )

```

Listing 5: Flusso end-to-end: regressione ortogonale e rendering della mappa 2D.