TRIQS

FLATIRON INSTITUTE
Center for Computational Quantum Physics

Ab initio DMFT: introduction to DFTTools, solid_dmft, and related tools

A. Hampel[1]

[1]Center for Computational Quantum Physics, Flatiron Institute, Simons Foundation

08/31/2023

ahampel@flatironinstitute.org

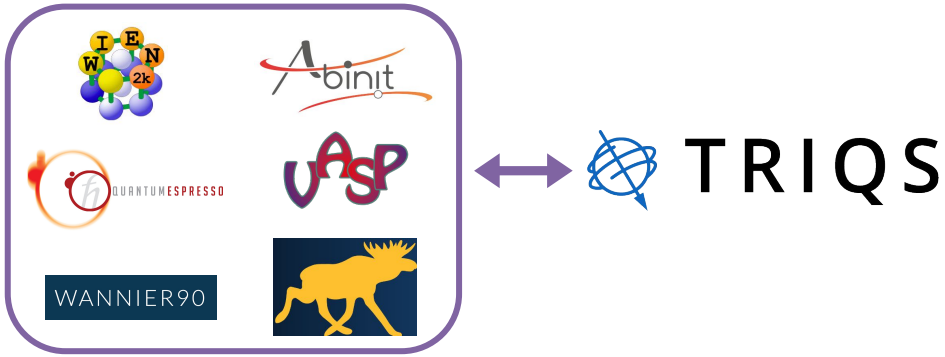
Outline

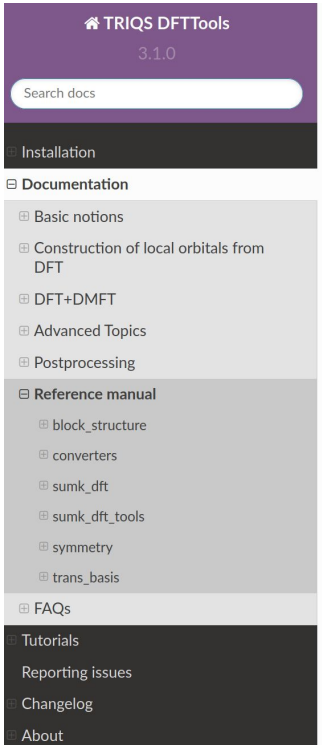
1. TRIQS/DFTTools: connection to ab initio codes
2. TRIQS/solid_dmft: full DFT+DMFT wrapper
3. TRIQS: available impurity solvers
4. Analytic continuation with TRIQS
5. FermiSee: phenomenology & visualization
6. solid_dmft tutorial

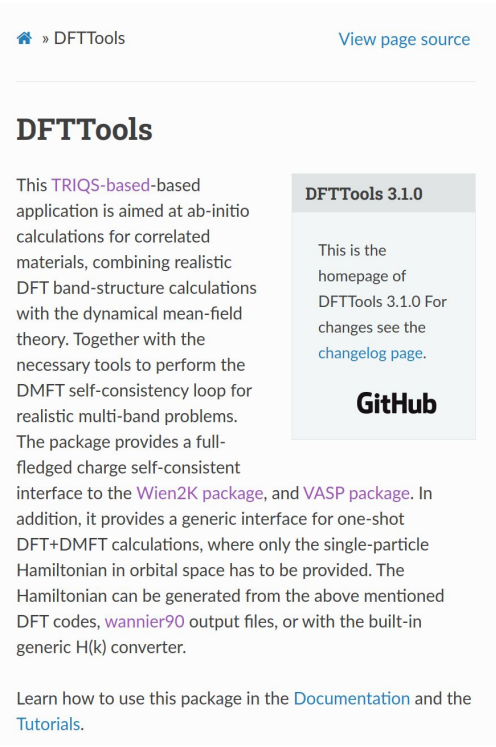
FLATIRON INSTITUTE

1. TRIQS/DFTTools: triqs.github.io/dft_tools

- same structure as TRIQS main project, i.e. automatic reference manual and tutorials
- issues and discussions on github.com/triqs/dft_tools
- M. Aichhorn *et al. CPC* '16 ~ 150 citations



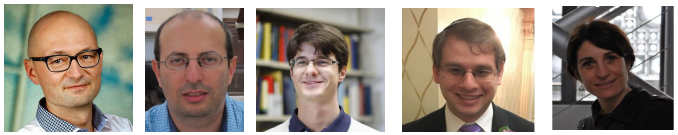




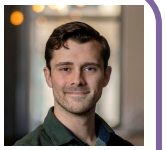
FLATIRON INSTITUTE

1. TRIQS/DFTTools: triqs.github.io/dft_tools

- same structure as TRIQS main project, i.e. automatic reference manual and tutorials
- issues and discussions on github.com/triqs/dft_tools
- M. Aichhorn *et al. CPC* '16
  ~ 150 citations


M. Aichhorn L. Pourovski M. Zingl J. Karp V. Vildosola


S. Beck M. Merkel


O. Gingras


Elk A. James

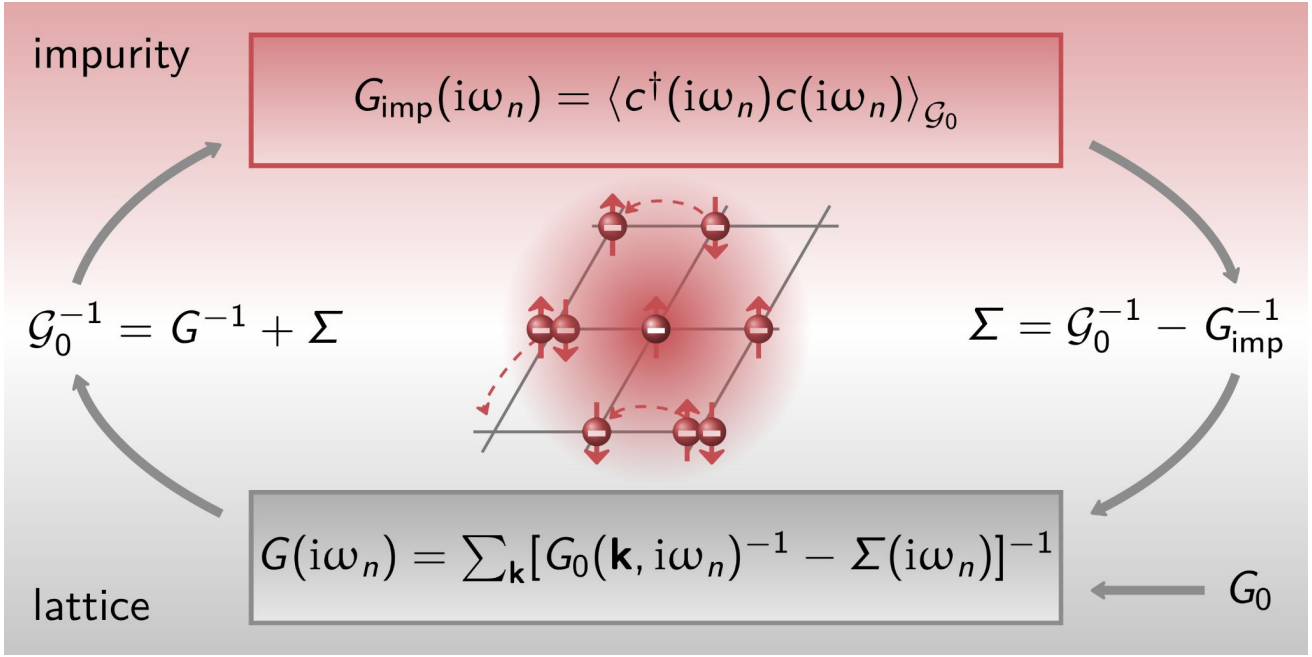

A. Hampel O. Peil T. Wehling

TRIQS

FLATIRON INSTITUTE

1. TRIQS/DFTTools: electronic structure interface

$$\hat{G}(\mathbf{k}, \mathrm{i}\omega_n) = \sum_{\nu\nu'} \left[\mathrm{i}\omega_n + \mu - \hat{\epsilon}(\mathbf{k}) - \Delta\hat{\Sigma}(\mathbf{k}, \mathrm{i}\omega_n) \right]^{-1}_{\nu\nu'} |\phi_{\nu\mathbf{k}}\rangle \langle\phi_{\nu'\mathbf{k}}|$$



Adapted from S.Beck

1. TRIQS/DFTTools: core functionality

- `lattice_gf()` $G(\mathbf{k},\omega)$
- `extract_G_loc()` $G(\omega)=\sum_k G(\mathbf{k},\omega)$
- `downfold / upfold` $P_{m\nu R}(\mathbf{k})$
- `calc_mu()` μ
- `calc_dc()` $\mathbf{\Sigma}^{DC}$
- `blockstructure class: analyse_block_structure_from_gf()`

FLATIRON INSTITUTE

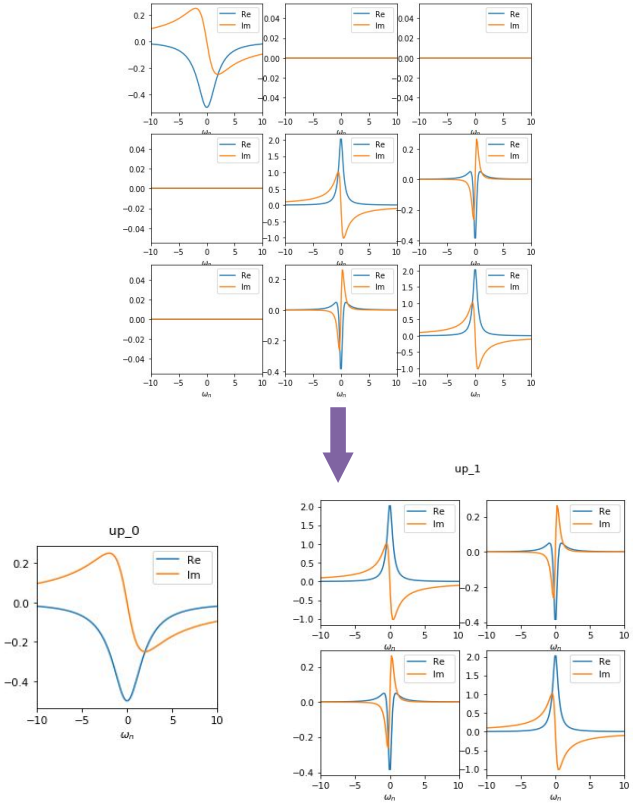
1. DFTTools example: Wannier90 converter + basics

```
from triqs_dft_tools.sumk_dft import SumkDFT
from triqs_dft_tools.converters import Wannier90Converter
from triqs.gf import *

Converter = Wannier90Converter(seedname='svo_t2g')
Converter.convert_dft_input()

mesh = MeshImFreq(beta=40, S='Fermion', n_iw=1025)
sumk = SumkDFT(hdf_file='svo_t2g.h5', mesh=mesh)

sumk.calc_mu()

Gloc_iw = sumk.extract_G_loc()

sumk.analyse_block_structure_from_gf(Gloc_iw)

sumk.set_Sigma(Sigma)
```

FLATIRON INSTITUTE

1. DFTTools functionality: SumkDFT

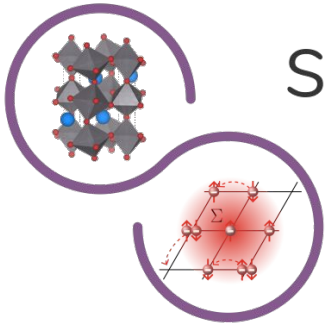
- tutorials for DMFT calculation with Wien2k, Elk, Vasp, and QE / Wannier90
- post-processing:
  - spectral function: `density_of_states()`, `spaghettis()`
  - Fermi surfaces: `spectral_contours()`
  - transport / optical conductivity $\sigma(\Omega)$
- manipulate orbital structure with `blockstructure` class (right)
- charge self-consistency updates of ρ



FLATIRON INSTITUTE

2. solid_dmft: DFT+DMFT calculations

- TRIQS flagship implementation of DFT+DMFT
- Scalability with scriptable config file
- interface to Vasp and Quantum Espresso for CSC calculations [2]
- Reproducibility: versioning, h5 storage, convergence metrics
- Flexible solver choice: cthyb, ctseg, ctint, FTPS, Hubbardl, Hartree, …
- Online documentation & tutorials: triqs.github.io/solid_dmft



solid_dmft

A versatile python wrapper to perform DFT + DMFT calculations utilizing the TRIQS software library.


S. Beck


M. Merkel (ETH)

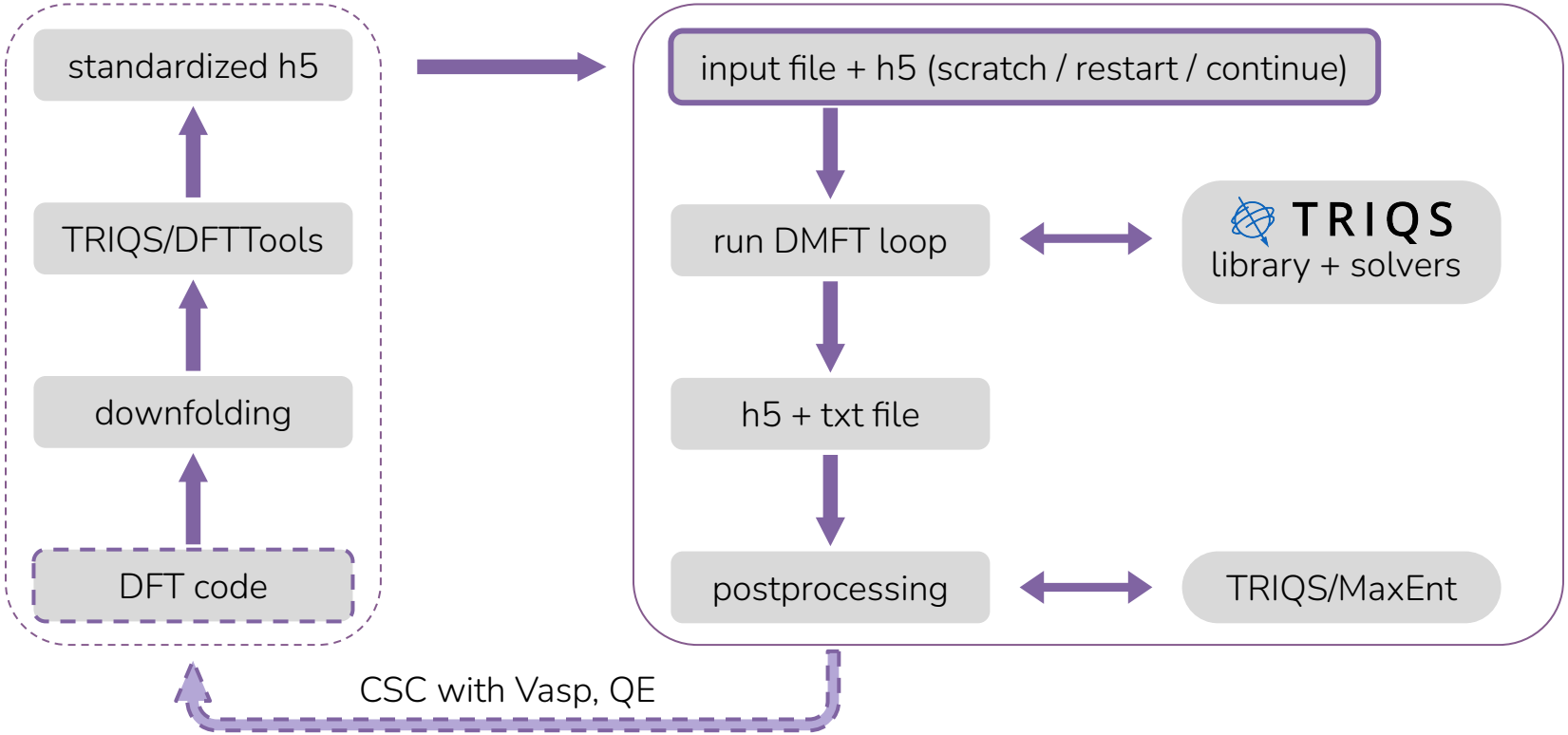

A. Carta (ETH)


A. Hampel

[1] M. Merkel, A. Carta, S. Beck, AH, JOSS, doi.org/10.21105/joss.04623 (2022)
[2] S. Beck, AH, O. Parcollet, C. Ederer, and A. Georges, JoP: Condensed Matter, 34 (2022)

FLATIRON INSTITUTE

2. solid_dmft: workflow



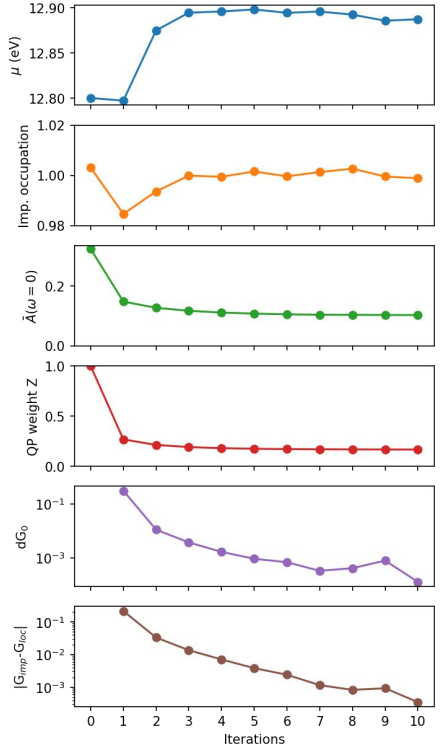
FLATIRON INSTITUTE

2. solid_dmft: example & tutorials

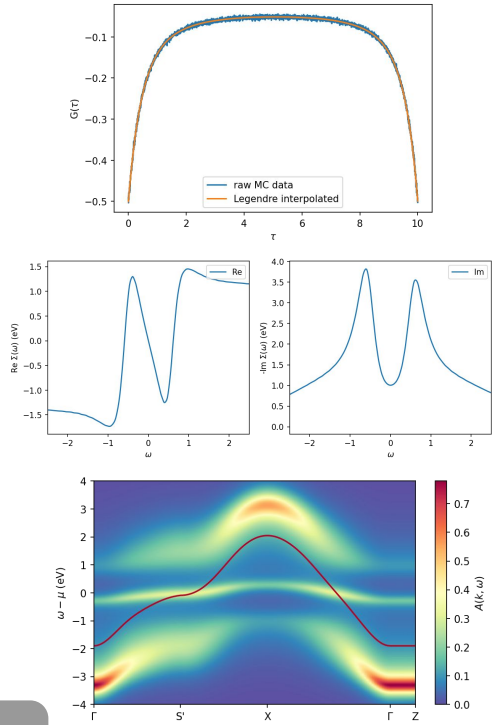
example config file:

```
[general]
seedname = SVO
beta = 10
prec_mu = 0.001

solver_type = cthyb
n_l = 35

h_int_type = kanamori
U = 8.0
J = 0.65

n_iter_dmft = 10
g0_mix_type = linear
g0_mix = 0.9

dc_type = 1
dc = True
dc_dmft = False

[solver]
length_cycle = 120
n_warmup_cycles = 1e+4
n_cycles_tot = 1e+8
measure_G_l = True
```

mpirun solid_dmft



postprocessing



github.com/TRIQS/tutorials/AbinitioDMFT
more tutorials: triqs.github.io/solid_dmft/tutorials

FLATIRON INSTITUTE

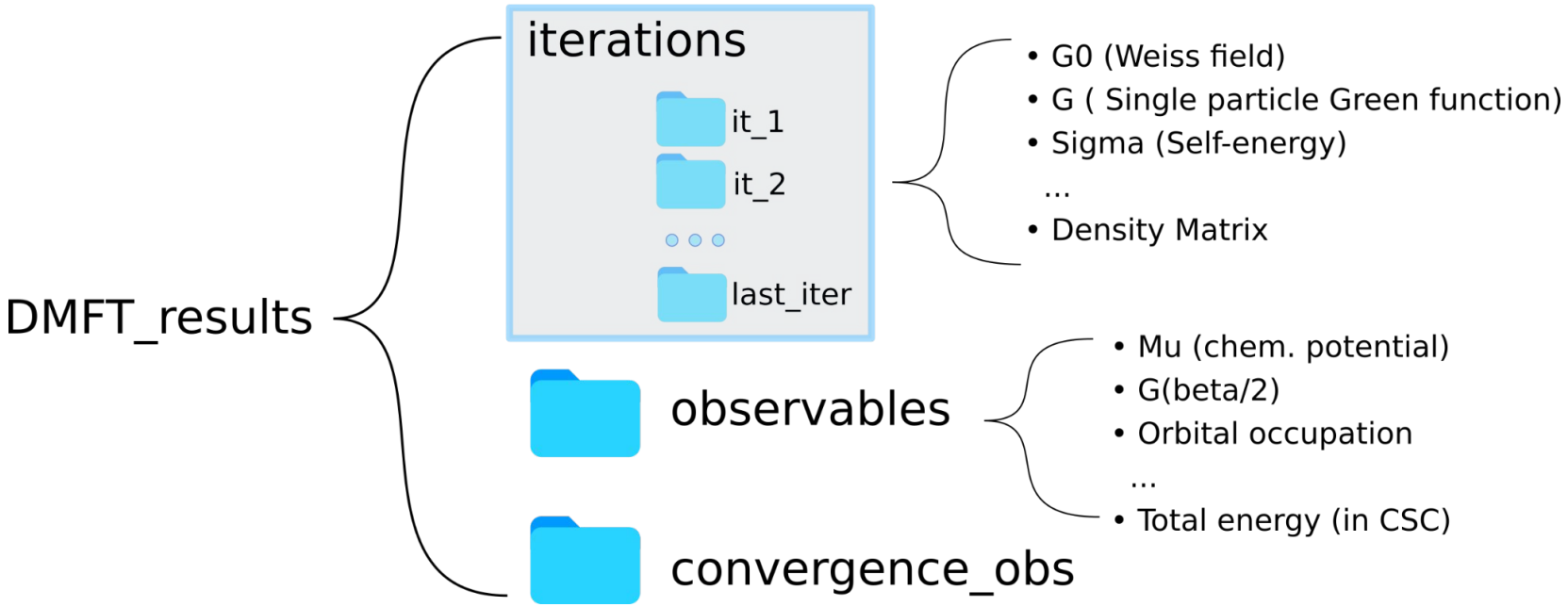
2. solid_dmft: input

- divided into 4 sections:
  - general
  - solver
  - dft
  - advanced
- extensive tutorials (including charge self-consistency) on triqs.github.io/solid_dmft/tutorials

```
[general]
seedname = vasp
csc = True
solver_type = cthyb
beta = 10
n_iter_dmft_first = 5
n_iter_dmft = 12

h_int_type = kanamori
U = 6.5
J = 0.65
dc_type = 1
dc_dmft = True
calc_energies = True

[solver]
length_cycle = 120
n_warmup_cycles = 8000
n_cycles_tot = 1e+6
measure_density_matrix = True

[dft]
dft_code = vasp
dft_exec = vasp_std
n_cores = 1
```

FLATIRON INSTITUTE

2. solid_dmft: standardized output for reproducibility



standardized output to continue previous calculations or just load self-energy

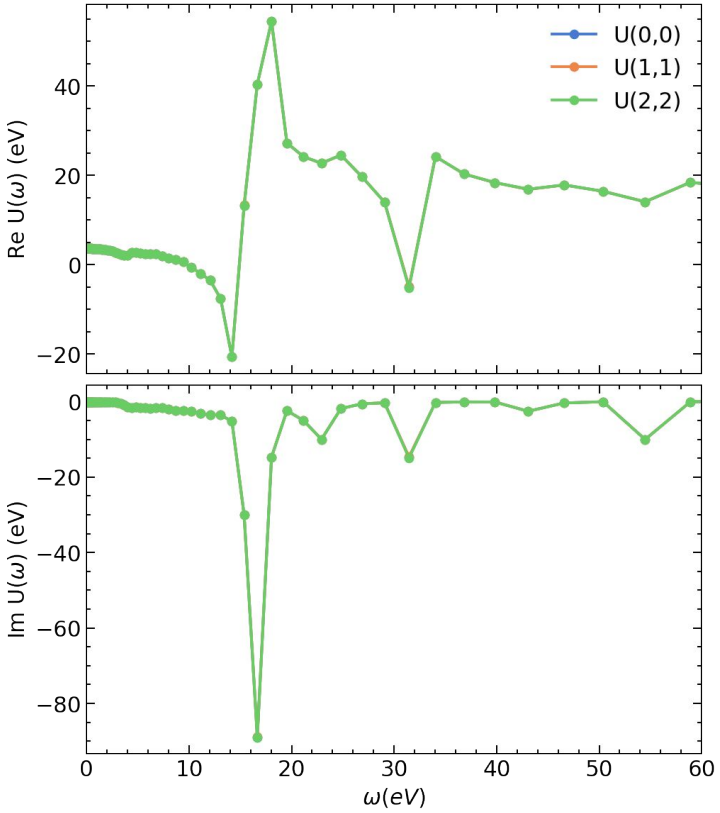
FLATIRON INSTITUTE

2. solid_dmft utils: cRPA reader

- RESPACK results reader

```
from solid_dmft.postprocessing.eval_U_cRPA_RESPACK
import read_interaction

RP = read_interaction(seed='svo',
                      path='./cRPA')

for key, value in RP.__dict__.items():
    print(key)
```

- gives access to numpy arrays of:
  - U_R, V_R, J_R, X_R
  - Uijkl, Vijkl
  - Uij_w, Jij_w
- same for Vasp cRPA
- results can be used in DMFT run*

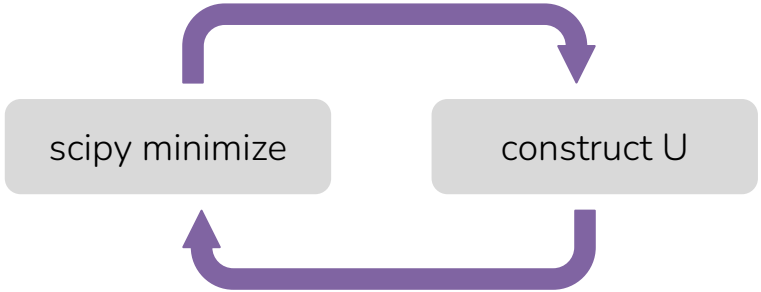


FLATIRON INSTITUTE

2. solid_dmft utils: parameterize Coulomb tensor

- module `eval_U_cRPA_Vasp.py`:

$$\hat{H}_{\text{int}} = \frac{1}{2} \sum_{\sigma\sigma'} \sum_{mm'm''m'''} U_{mm'm''m'''} \; c^{\dagger}_{m\sigma} c^{\dagger}_{m'\sigma'} c_{m'''\sigma'} c_{m''\sigma}$$



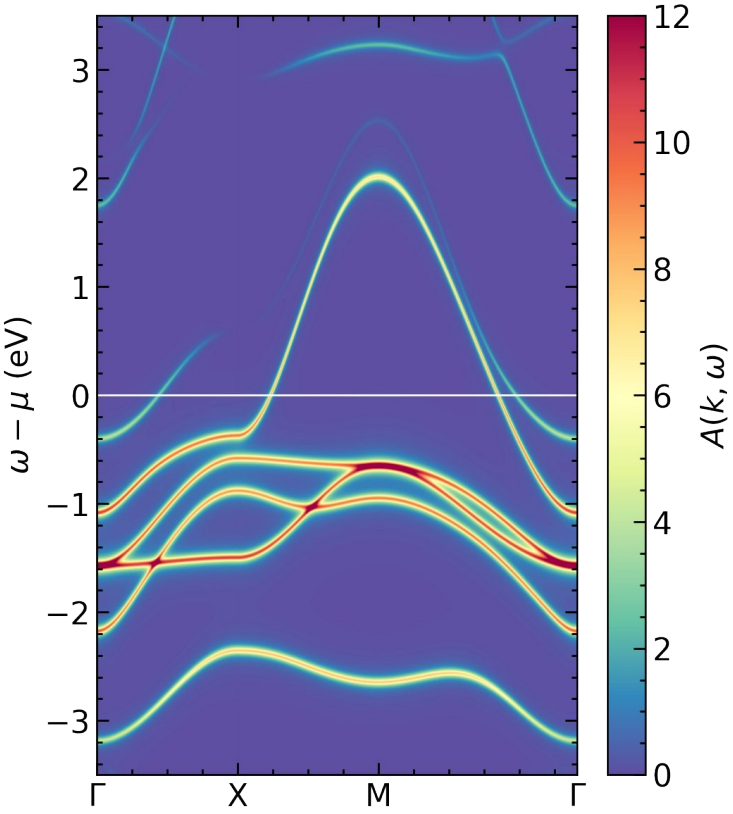
- `fit_kanamori(uijkl, n_orb, fit_2, fit_3, fit_4)`
- `fit_slater_fulld(u_ijij, u_ijji, U_init, J_init, fixed_F4_F2)`

$$\begin{aligned}\hat{H}_{\text{kan}} = & \frac{1}{2} \sum_{\sigma} \sum_{m} \mathcal{U} \; \hat{n}_{m\sigma} \hat{n}_{m\bar{\sigma}} \\ & + \frac{1}{2} \sum_{\sigma} \sum_{m \neq m'} \left[\mathcal{U}' \; \hat{n}_{m\sigma} \hat{n}_{m'\bar{\sigma}} + (\mathcal{U}' - \mathcal{J}) \hat{n}_{m\sigma} \hat{n}_{m'\sigma} \right] \\ & + \frac{1}{2} \sum_{\sigma} \sum_{m \neq m'} \left[\mathcal{J} \; c^{\dagger}_{m\sigma} c^{\dagger}_{m'\bar{\sigma}} c_{m\bar{\sigma}} c_{m'\sigma} + \mathcal{J}_C \; c^{\dagger}_{m\sigma} c^{\dagger}_{m\bar{\sigma}} c_{m'\bar{\sigma}} c_{m'\sigma} \right]\end{aligned}$$

FLATIRON INSTITUTE

2. solid_dmft post-processing with H(*R*)

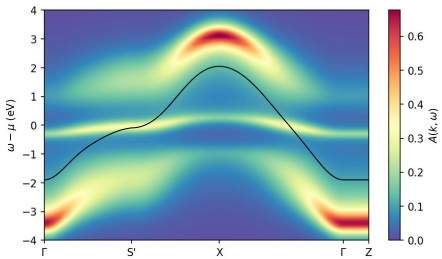
- leverage Wannier interpolation to calculate G(k, ω)
- no pre-computing of projectors on k-path necessary
- high precision mesh allows to find contours and QP dispersion
- loads automatically self-energy and other parameters from solid_dmft h5



FLATIRON INSTITUTE

2. solid_dmft post-processing with H(*R*)

```
from solid_dmft.postprocessing import plot_correlated_bands as pcb

w90_dict = {'w90_path': './data/mlwf/', 'w90_seed': 'lco',
            'n_orb': 1, 'mu_tb': 12.7367}

tb_dict = {'bands_path': [('G', 'S\''), ('S\'', 'X'), ('X', 'G'), ('G', 'Z')],
           'n_k': 50, 'G': [ 0.0,  0.0,  0.0],
           …}

sigma_dict = {'dmft_path': 'b10-U3.6/lco_wannier.h5',
              'it': 'last_iter', 'spin': 'up', 'block' : 0 }

tb_bands, alatt_kw, freq = pcb.get_dmft_bands(with_sigma='calc',
                                              **w90_dict, **tb_dict, **sigma_dict)

fig, ax = plt.subplots(1)
pcb.plot_bands(fig, ax, alatt_kw, tb_bands, freq, n_orb=w90_dict['n_orb'],
               alatt=True, colorscheme_bands='Greys', colorscheme_alatt='Spectral_r')
```



FLATIRON INSTITUTE

3. TRIQS: available impurity solvers

solver name	method	# orb	measure	comments
cthyb	ct-qmc hyb expansion	~5	$G(\tau)$, $G^{(2)}$, χ_{AB}, ρ_{imp}	small U/Δ, off diag Δ → sign
ctseg	ct-qmc hyb segment picture	~8	$G(\tau)$, $G^{(2)}$, χ_{AB}, ρ_{imp}	small U/Δ, nn int only, $U(\tau)$
ctint	ct-qmc interaction expansion	~80	$G(\tau)$, $G^{(2)}$, χ_{AB}	small Δ/U, nn int only, $U(\tau)$
forkTPS	fork tensor product states	~5	$G(t)$, χ_{AB}	η~1e-2 , Kanamori only
hartree_fock	Hartree / Hartree-Fock	~20	$\sum^{HF}$	no ω, HF only
hubbardI	ED of impurity problem	~7	$G(i\omega_n)$, $G(\omega)$, $G^{(2)}$, χ_{AB}	Δ>0
nrgljubljana_interface	NRG	~3	$G(\omega)$, $A(\omega)$, χ_{AB}	log mesh around ω=0
pomerol2triqs	ED of impurity problem	~3	$G(\omega)$, $G(i\omega_n)$, $G^{(2)}$, χ_{AB}	discretized bath
w2dynamics_interface	ct-qmc hyb / seg exp + worm	~5	$G(\tau)$, $G^{(2)}$ + worm, χ_{AB}, ρ_{imp}	small U/Δ, off diag Δ → sign, $U(\tau)$

4. analytic continuation: TRIQS

- TRIQS provides several apps to analytically continue Matsubara Green functions:

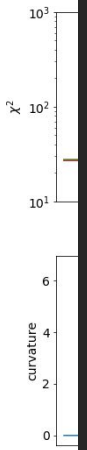
$$G(i\omega_n) = \int_{-\infty}^{\infty} \frac{d\omega}{2\pi} \frac{1}{i\omega_n - \omega} A(\omega) \quad \text{or} \quad \mathbf{G} = \mathbf{K}\mathbf{A} \text{ (matrix form)}$$

- triqs.github.io/maxent
  - G. J. Kraberger et al. , PRB 96 (2017)
- triqs.github.io/Nevanlinna
  - S. Iskakov et al. , CPC (2023)
- triqs.github.io/omegamaxent_interface
  - ΩMaxEnt code, D. Bergeron and A.-M.S. Tremblay, PRE 94 (2016)
- krivenko.github.io/som
  - Stochastic Optimization Method, I. Krivenko et al. , CPC 239 (2019)

FLATIRON INSTITUTE

4. analytic continuation: TRIQS/maxent

- triqs.github.io/maxent
- different ways to choose α implemented:
  - line-fit
  - from curvature of log(χ^2) vs log(α)
  - Bryan
- matrix valued continuation
- self-energy continuation (element-wise)




```
G_iw = GfImFreq(beta=10, indices=[0])
G_iw << SemiCircular(1)-0.5*SemiCircular(0.5)
G_tau = GfImTime(beta=10, indices=[0],
n_points=2501)
G_tau.set_from_fourier(G_iw)
G_tau.data[:, 0, 0] += 1.e-5 *
np.random.randn(len(G_tau.data))

from triqs_maxent import *
tm = TauMaxEnt(cost_function='bryan',
probability='normal')
tm.set_G_tau(G_tau)
tm.set_error(1e-4)
# run maxent
result = tm.run()
result.get_A_out('LineFitAnalyzer')
```

FLATIRON INSTITUTE

4. analytic continuation: solid_dmft + maxent

- integration between solid_dmft and maxent (block structure, DC, μ)
- MPI parallelized over blocks
- continuation of:
  - G_{imp}: `postprocessing.maxent_gf_imp`
  - G_{latt}: `postprocessing.maxent_gf_latt`
  - $\sum_{imp}$: `postprocessing.maxent_sigma`
- writes result back to h5
- automatically used by post-processing routines

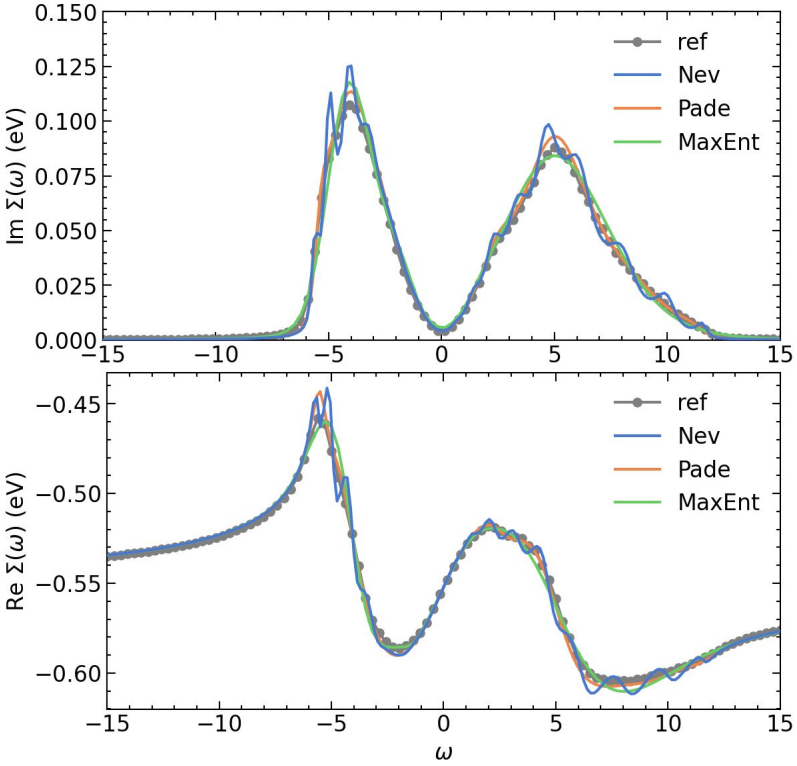
```
from solid_dmft.postprocessing import maxent_sigma

# use pcb maxent script to continue self energy
Sigma_real_freq =
maxent_sigma.main(external_path=h5_file,
                  omega_min=-10, omega_max=10,
                  maxent_error=0.03,
                  iteration=None,
                  n_points_maxent=101,
                  n_points_alpha=50,
                  analyzer='LineFitAnalyzer',
                  n_points_interp=2001,
                  n_points_final=1001,
                  continuator_type='inversion_dc')
```

FLATIRON INSTITUTE

4. analytic continuation: TRIQS/nevanlinna & Pade

- Nevanlinna, matrix valued Caratheodory, and Hardy optimizations
- works best for non-continuous spectra
- further extension necessary for noisy data
- triqs.github.io/Nevanlinna/latest/documentation

1 band GW self-energy example:



FLATIRON INSTITUTE

5. FermiSee: WebApp for data visualization

$$A(\omega, \mathbf{k}) = -\frac{1}{\pi}\mathrm{Im} \sum_{\alpha=\alpha'} [\omega + \mu - \epsilon(\mathbf{k}) - \Sigma(\omega)]^{-1}_{\alpha\alpha'}$$

🔬 fermisee.flatironinstitute.org

🧰 github.com/TRIQS/FermiSee

👷 testers and developers welcome

TRIQS

WANNIER90 PythTB


S. Beck

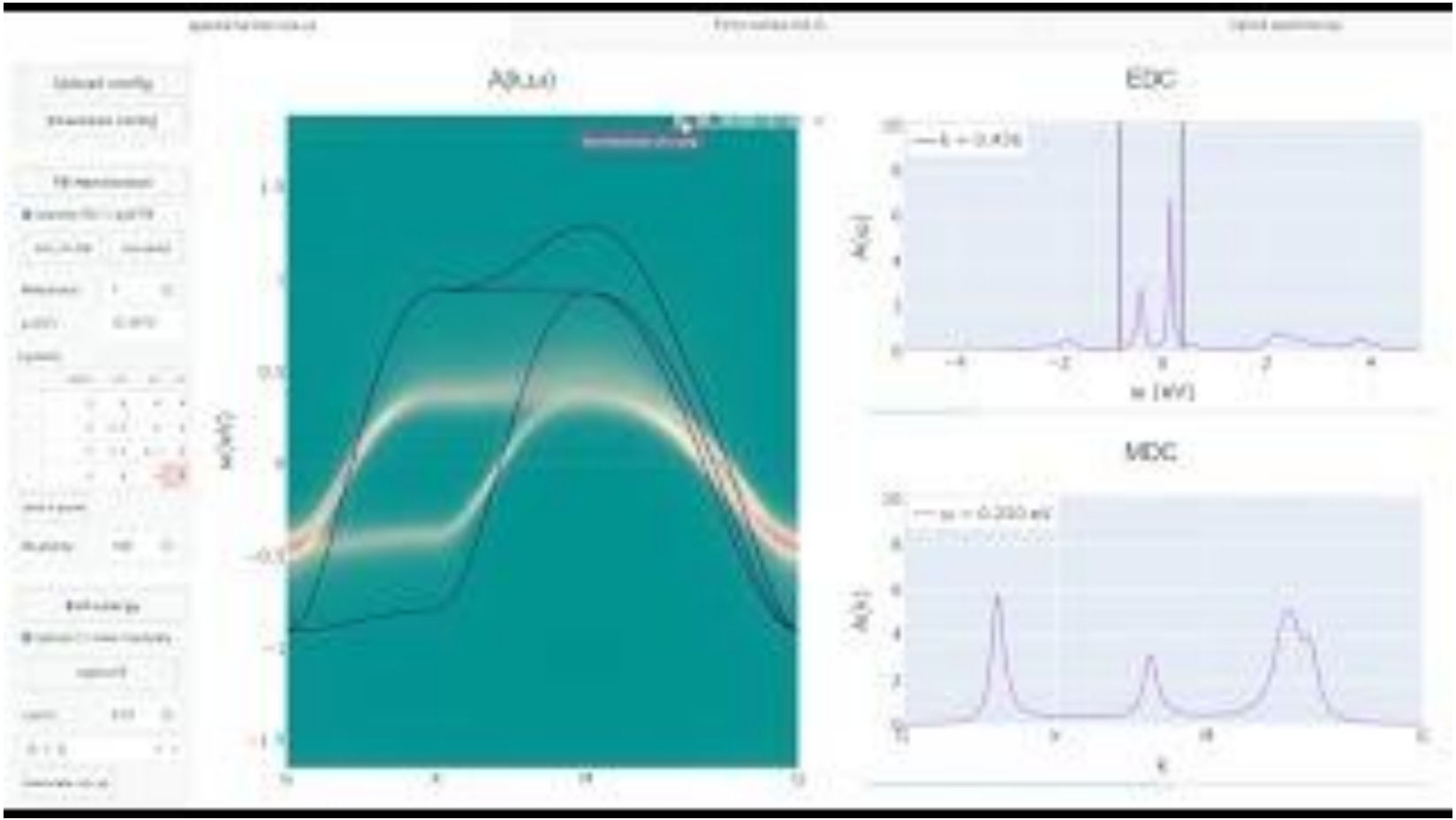

S. Rahim


A. Hampel

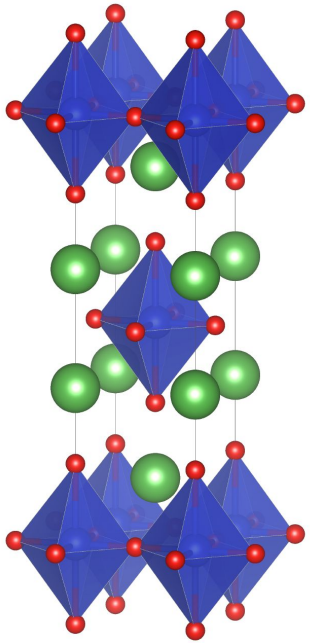
plotly | Dash

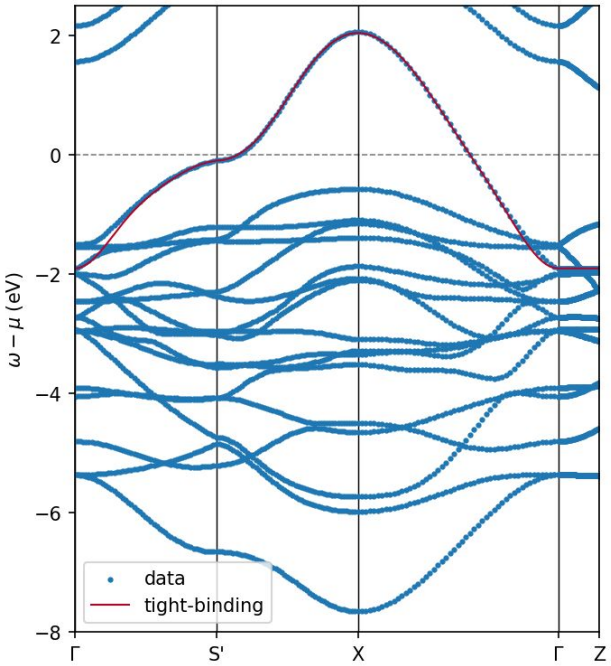
FLATIRON INSTITUTE

5. FermiSee demo

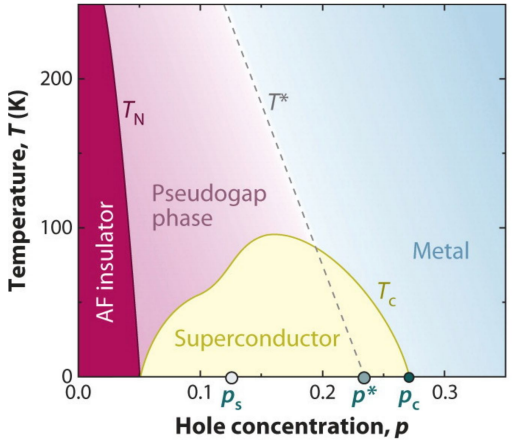


6. solid_dmft tutorial: Mott insulator La_2CuO_4





- simple 1-band model, 2D square-lattice sheets
- $1e^-$ per Cu atom, mainly d_{x2-y2} character



L. Taillefer, Ann. Rev. Condens. Matter Phys. 1, 51 (2010)

FLATIRON INSTITUTE

Summary

TRIQS

solid_dmft

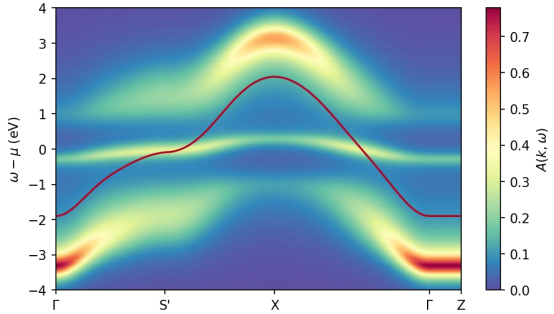
A versatile python wrapper to perform DFT + DMFT calculations utilizing the TRIQS software library.

TRIQS ecosystem to perform ab-initio simulations for correlated electron systems

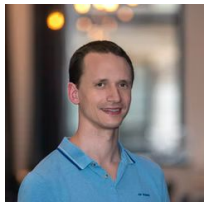
solid_dmft as flagship implementation for DFT+DMFT and embedding

Analytic continuation packages

FermiSee: webapp for data visualization



Acknowledgements:


N. Wentzell


S. Beck


O. Parcollet


A. Georges

FLATIRON INSTITUTE