Minimal pole method for spectral functions

Lei Zhang
University of Michigan
lzphy@umich.edu

J. Chem. Phys. 162, 214111 (2025), L. Zhang et al.

Introduction — Spectral function

Spectral function (measured by ARPES):

$$A(\omega) = \sum_l A_l^{(\mathrm{real})}\delta(\omega - \xi_l^{(\mathrm{real})}) \text{ with } A_l \text{ (fermion) or } \mathrm{sgn}(\omega)A_l \text{ (boson) being positive}$$

Finite system: a finite number of delta peaks

Infinite system: infinite delta peaks with infinitesimal weights -> broadened peaks

Connect with imaginary axis:

$$G^{\mathrm{Mat}}(i\omega_n) = \int_{-\infty}^{+\infty} d\omega \frac{A(\omega)}{i\omega_n - \omega} \text{ and } G(\tau) = \frac{1}{\beta}\sum_{n=-\infty}^{+\infty} e^{-i\omega_n\tau}G(i\omega_n)$$

Connect with real axis:

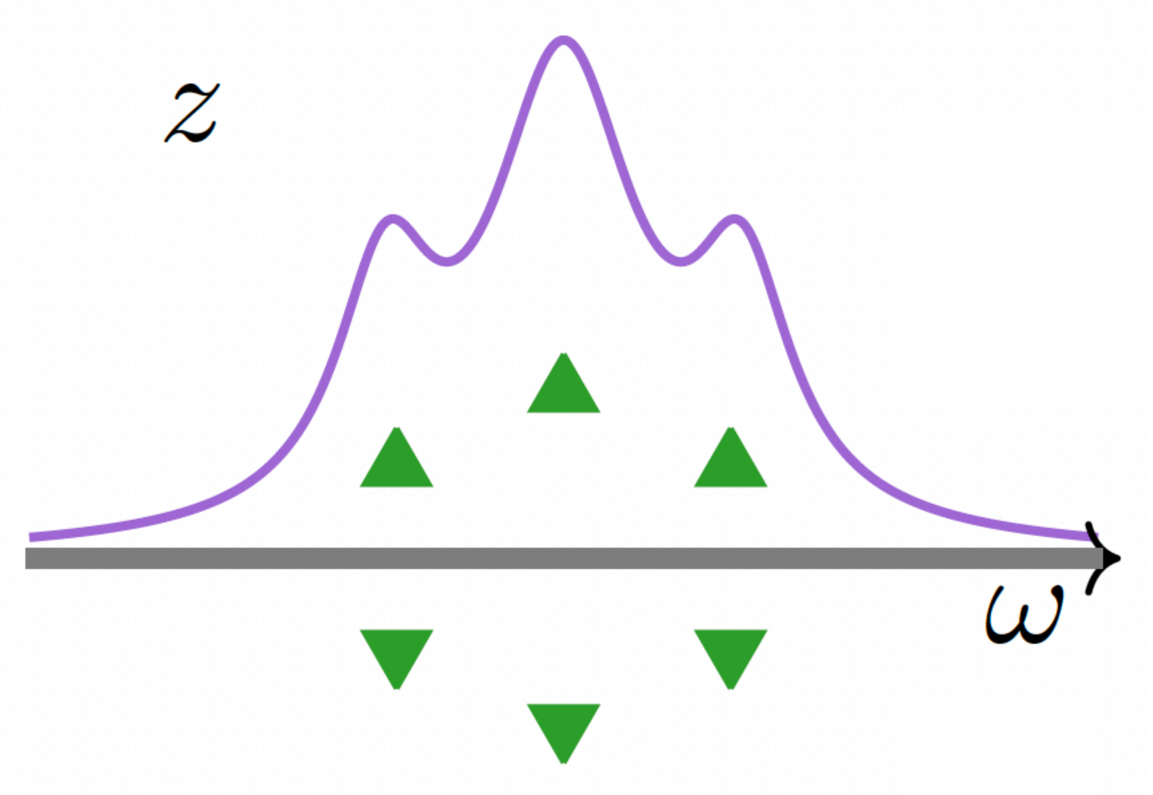
$$\mathrm{Im}[G^{\mathrm{Ret}}(\omega)] = -\pi A(\omega),\ \mathrm{Re}[G^{\mathrm{Ret}}(\omega)] = -\mathscr{H}[\mathrm{Im}[G^{\mathrm{Ret}}(\omega)]] \text{ and } G^{\mathrm{Ret}}(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} dt e^{-i\omega t}G^{\mathrm{Ret}}(\omega)$$

$$\text{BCF: } C(t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} d\omega e^{-i\omega t}A(\omega)\left[\coth\left(\frac{\beta\omega}{2}\right) + 1\right]$$

Introduction — Complex pole representation

Ansatz:

$$A(z) = \sum_{l=1}^{M} \left(\frac{A_l^{(\mathrm{dn})}}{z - \xi_l^{(\mathrm{dn})}} + \frac{A_l^{(\mathrm{up})}}{z - \xi_l^{(\mathrm{up})}} \right) \text{ with } \xi_l^{(\mathrm{up})} = (\xi_l^{(\mathrm{dn})})^* \text{ and } A_l^{(\mathrm{up})} = (A_l^{(\mathrm{dn})})^*$$



With $z \in \mathbb{C}$ and $A(z = \omega) \approx A_{\mathrm{exact}}(\omega)$

Applications:

1. Imaginary axis: $G(z) = -2\pi i \sum_{l=1}^{M} \frac{A_l^{(\mathrm{dn})}}{z - \xi_l^{(\mathrm{dn})}}, \quad \mathrm{Im} z > 0$

   Hilbert transform; recover GF in the upper-half plane, e.g., $G(z = i\omega_n) = G^{\mathrm{Mat}}(i\omega_n)$

2. Real-time simulations, e.g., HEOM:

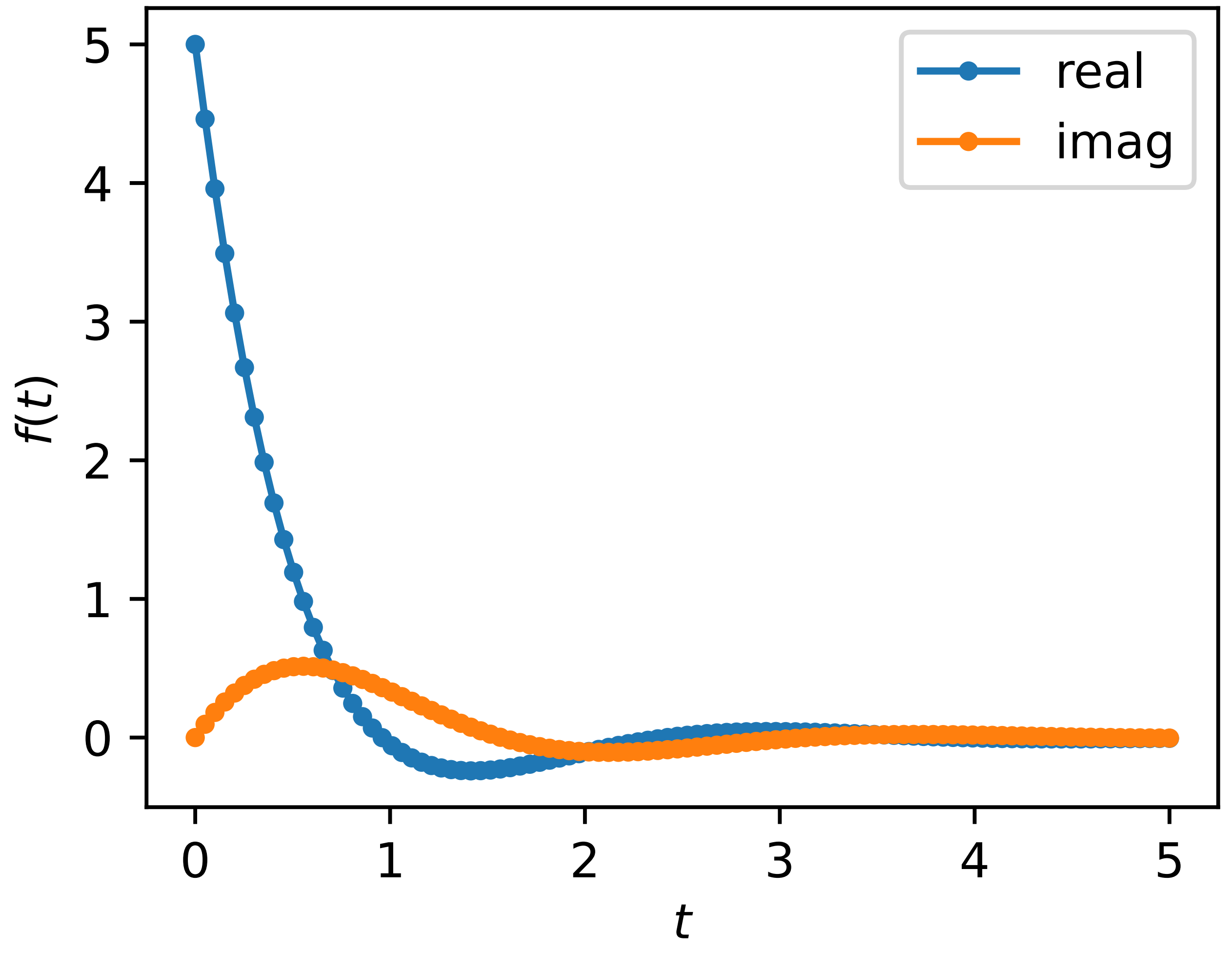
$$A(\omega) = J(\omega)\left[\coth\left(\frac{\beta\omega}{2}\right) + 1\right] \text{ -> } C(t) = \sum_{l=1}^{M} \eta_l e^{-\gamma_l t} \text{ with } \eta_l = -iA_l^{(\mathrm{dn})} \text{ and } \gamma_l = i\xi_l^{(\mathrm{dn})}$$

Introduction — Prony-like methods

Input: $\{f(t_k), t_k\}$ on a uniform grid

Ansatz: $f(t) \approx \sum_{i=1}^{M} R_i e^{s_i t} \xrightarrow{\text{discretize}} f(t_k) \approx \sum_{i=1}^{M} R_i z_i^k$

Goal: estimate $M \in \mathbb{Z}$, $R_i \in \mathbb{C}$ and $z_i \in \mathbb{C}$



Solution: ESPRIT, Matrix Pencil, Prony approximation…

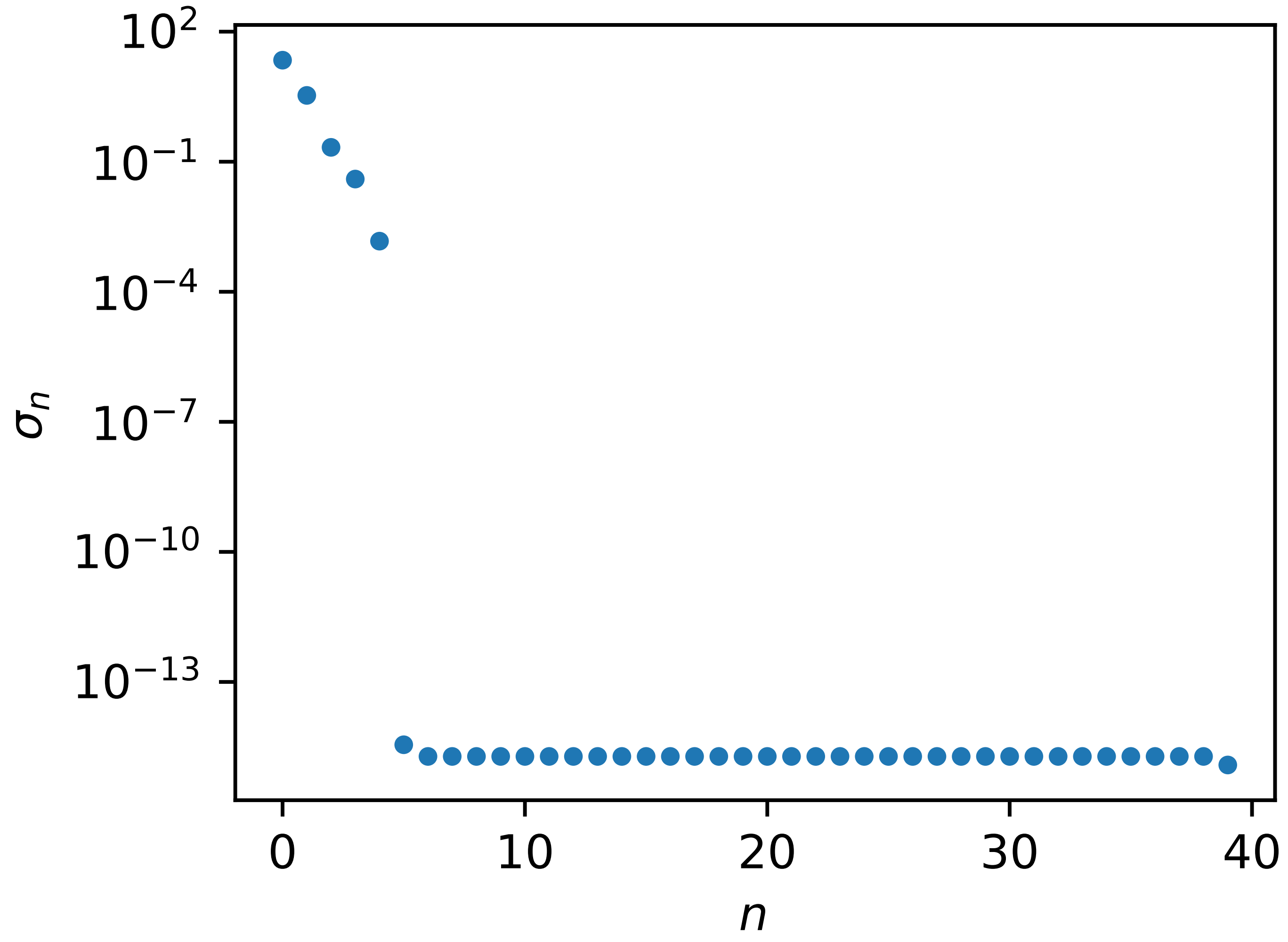
IEEE Trans. Acoust. Speech, Signal Process. 37, 984 (1989)

IEEE Trans. Acoustics, Speech, Signal Process. 38, 814 (1990)

Appl. Comput. Harmonic Anal. 19, 17 (2005)

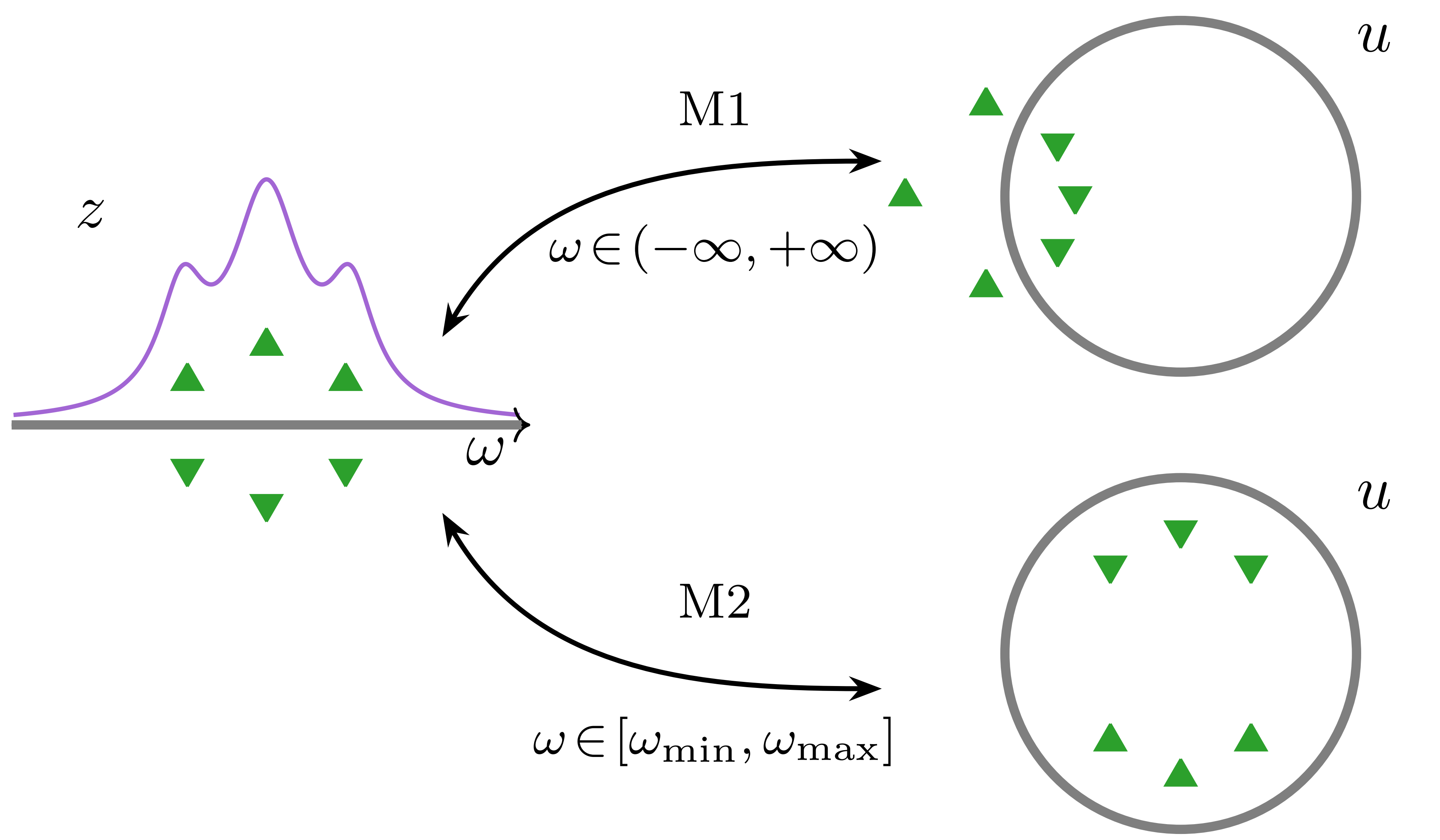
ESPRIT:

1. Construct $(N-L)\times(L+1)$ Hankel matrix $H_{ij} = f(t_{i+j})$, with $\frac{N}{3} \leq L \leq \frac{N}{2}$
2. Perform SVD: $H = U\Sigma W$
3. Obtain M from $\sigma_{M+1} < \varepsilon$
4. Obtain z_i from eigenvalues of $(W(1:M, 1:L)^T)^{+} W(1:M, 2:L+1)^T$
5. Obtain R_i from least-square fits



Robust to noise; use a minimal number of complex exponentials to fit the data to the given precision ε

Minimal Complex Pole Method (MPM)



Mapping 1 (M1):

$$\begin{cases} u &= f(z) &= \frac{z+i\omega_p}{z-i\omega_p} \\ z &= f^{-1}(u) &= i\omega_p \frac{u+1}{u-1} \end{cases}$$

Mapping 2 (M2):

$$\begin{cases} u &= f(z) &= z_\mathrm{s} + \sqrt{z_\mathrm{s}^2 - 1} \text{ with } z_\mathrm{s} = \frac{z-\omega_\mathrm{m}}{\Delta\omega_\mathrm{h}} \\ z &= f^{-1}(u) &= \frac{\Delta\omega_\mathrm{h}}{2}(u + \frac{1}{u}) + \omega_\mathrm{m} \end{cases}$$

Assume: analytic expression of $A(\omega)$ is known

Step 1 $A(z) = \sum_{l=1}^{M} \left(\frac{A_l^{(\mathrm{dn})}}{z - \xi_l^{(\mathrm{dn})}} + \frac{A_l^{(\mathrm{up})}}{z - \xi_l^{(\mathrm{up})}} \right) \xleftrightarrow[A'(u(z)) = A(z)]{} A'(u) = \sum_{l=1}^{M} \left(\frac{A_l'^{(\mathrm{dn})}}{u - \xi_l'^{(\mathrm{dn})}} + \frac{A_l'^{(\mathrm{up})}}{u - \xi_l'^{(\mathrm{up})}} \right) + \text{analytic part}$

Origin: Numerical analytic continuation L. Zhang et al., Phys. Rev. B 110, 035154 (2024) L. Zhang et al., Phys. Rev. B 110, 235131 (2024)

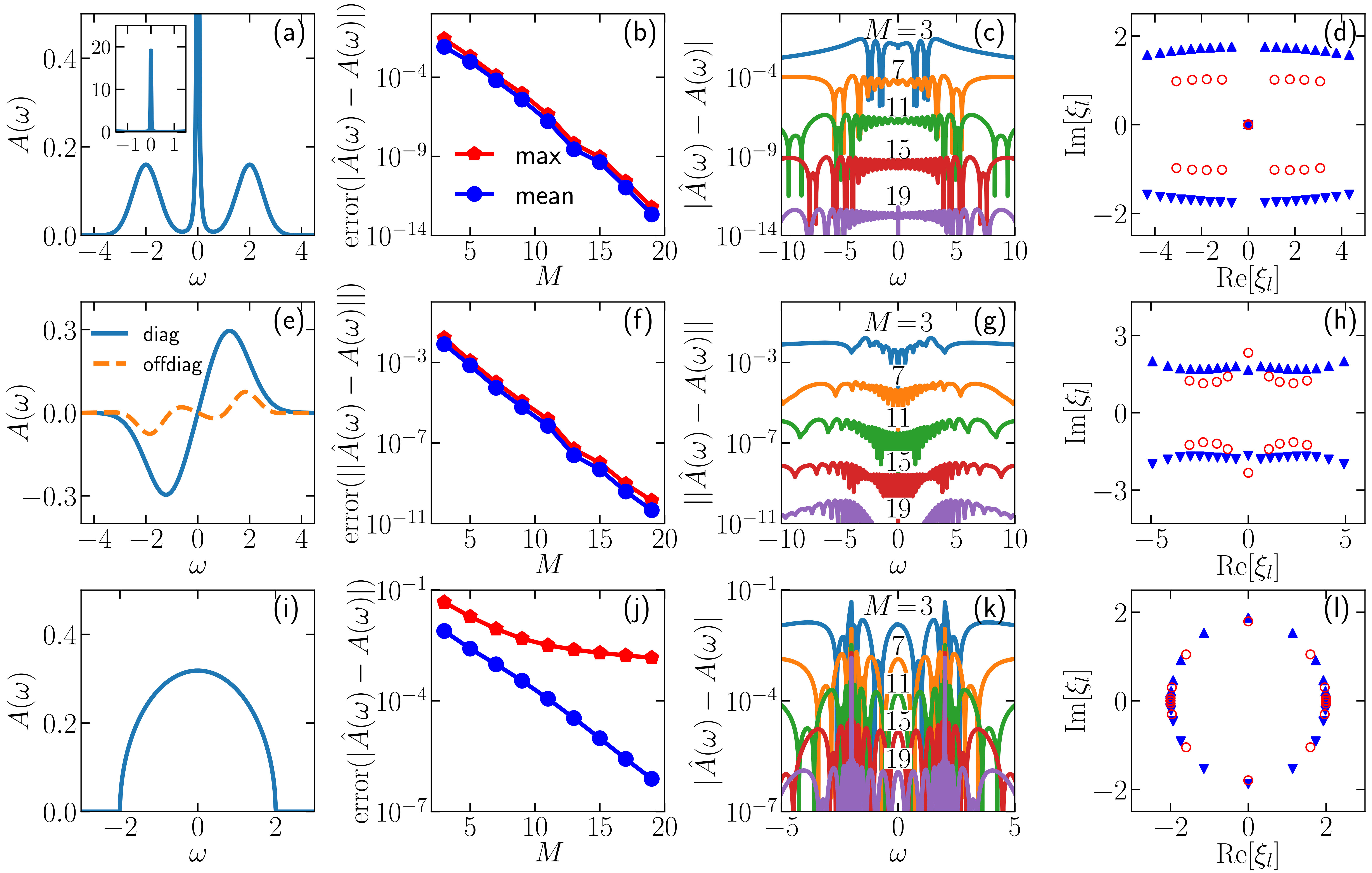
Step 2 Transformed moments $h_k := \frac{1}{2\pi i} \int_{\partial \bar{D}} du\, A'(u) u^k, \quad k \geq 0,$ $\quad h_k = \sum_l A_l' \xi_l'^k$

Imperfect input: utilize AAA, trapezoidal rule…

Step 3 ESPRIT to solve A_l' and ξ_l' and transform back to obtain A_l and ξ_l

Multiple orbitals: matrix-valued ESPRIT L. Zhang et al., Phys. Rev. B 110, 235131 (2024)

Results — spectral functions



(a) Kondo-like

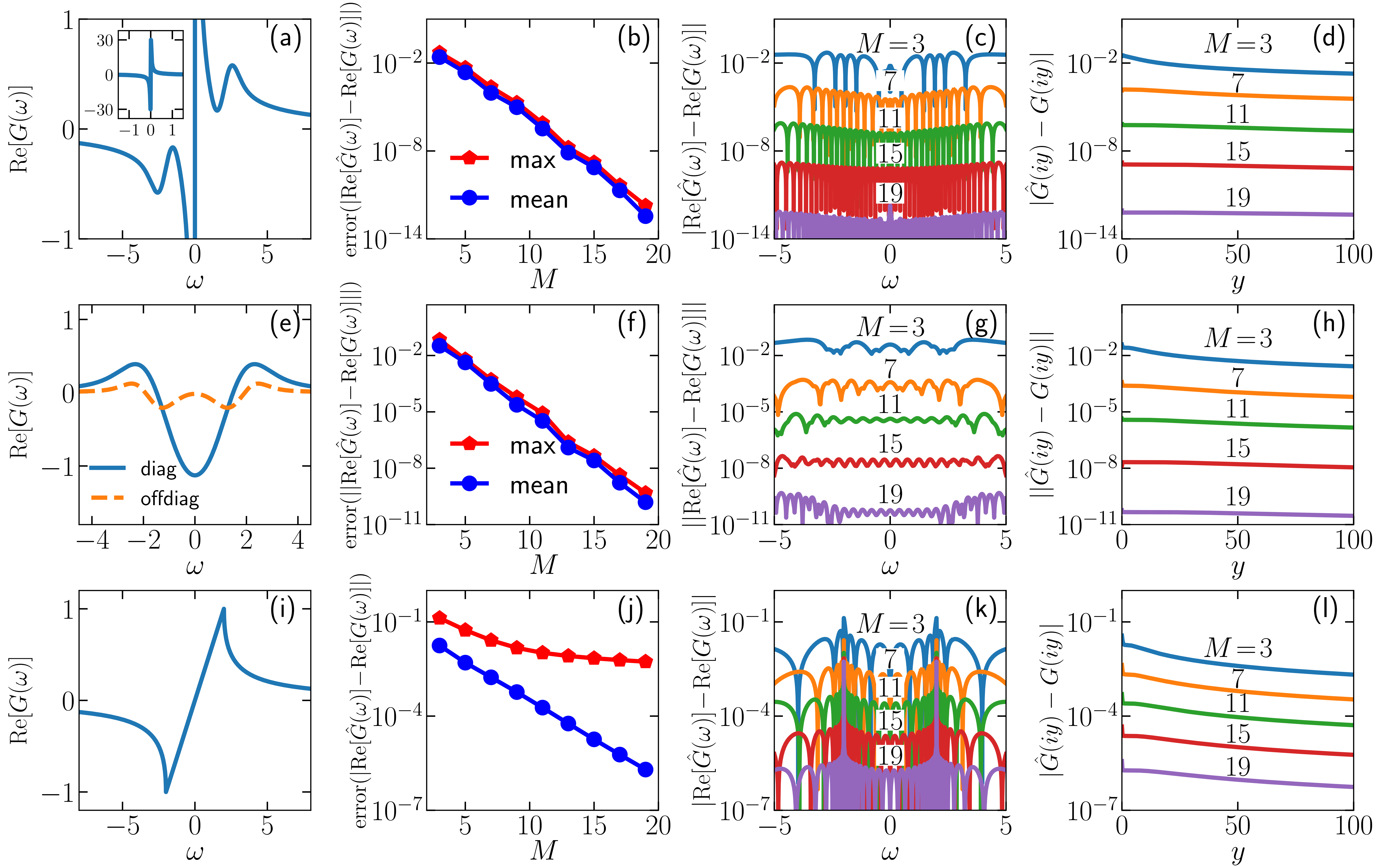
(b) Bosonic multi-orbital

(c) Semicircular

MPM (M1) was used

- Converges exponentially away from singularities
- Convergence slows down near the singularities

Results — recovery of GF's



(a) Kondo-like

(b) Bosonic multi-orbital

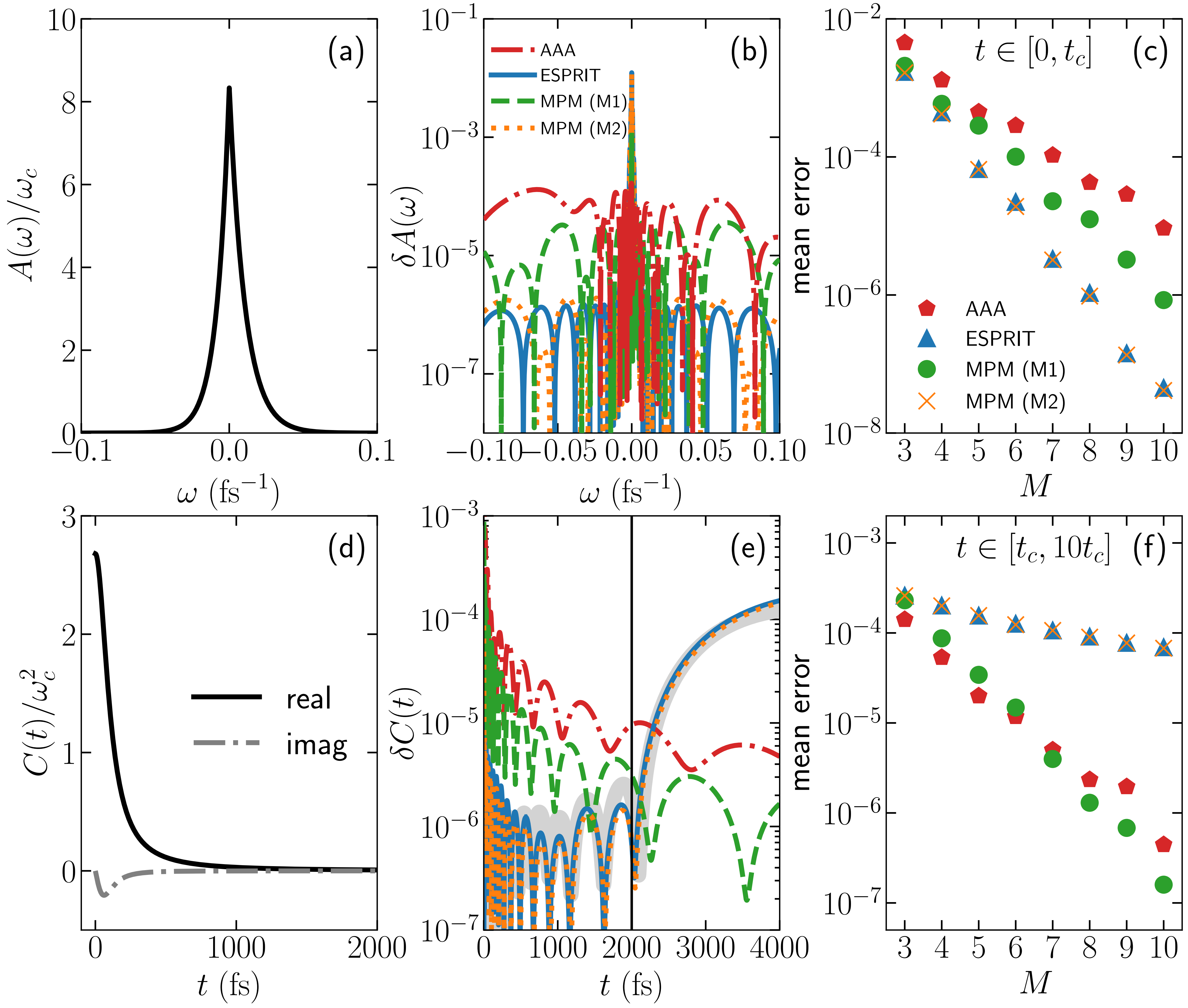
(c) Semicircular

MPM (M1) was used

- Converges exponentially away from singularities
- Convergence slows down near the singularities
- Computations based on complex poles are not affected by singularities

Results — BCF (Ohmic bath)

$T = 300K \ (\beta\omega_c \approx 0.240)$



$J(\omega > 0) = \omega e^{-\omega/\omega_c}$ with $J(\omega < 0) = -J(-\omega)$

$$A(\omega) = J(\omega)\left[\coth\left(\frac{\beta\omega}{2}\right) + 1\right]$$

AAA: logarithmic discretized grid on $\omega \in [-\omega_{\max}, \omega_{\max}]$

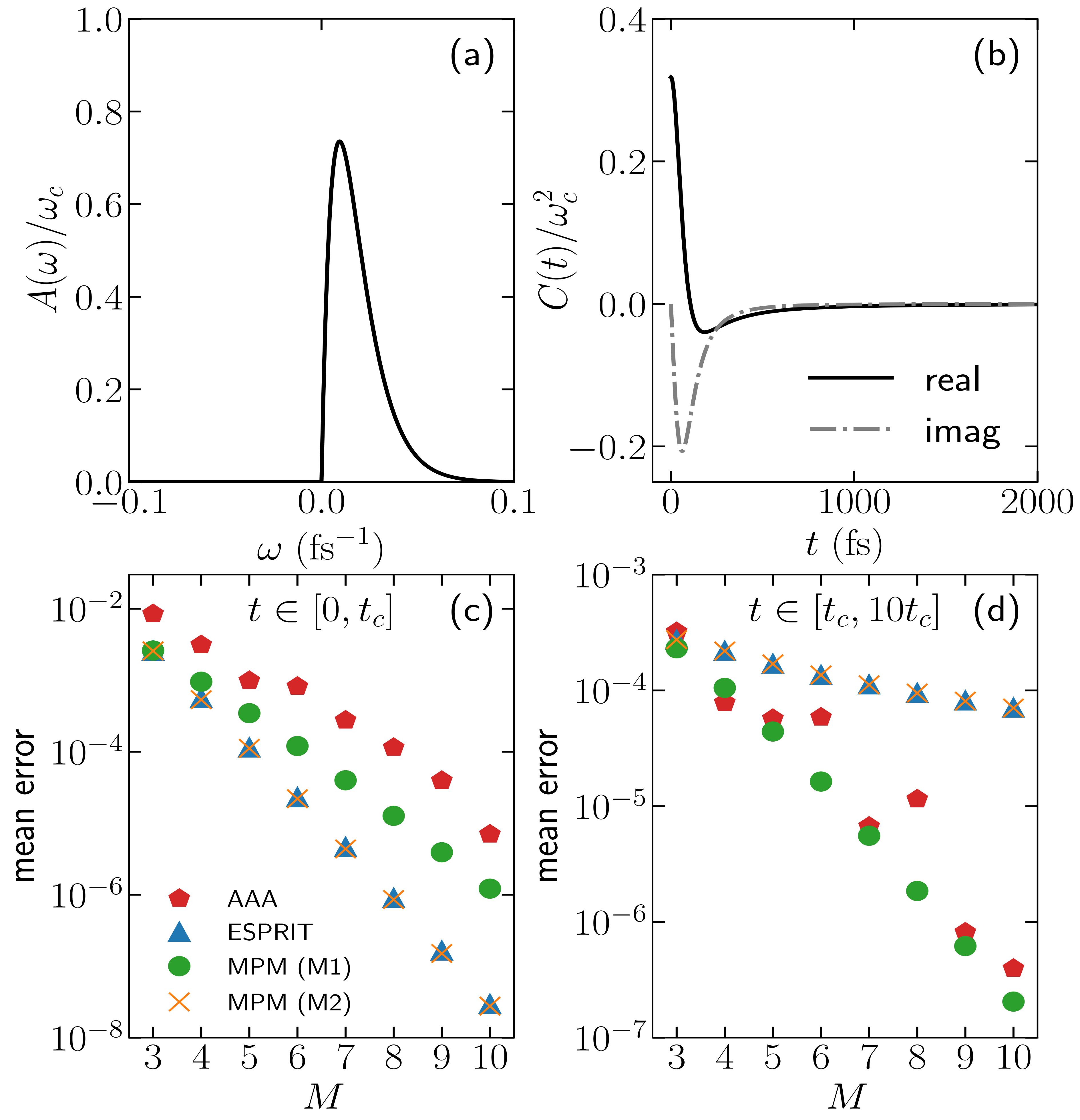
ESPRIT: uniform grid on $t \in [0, t_c]$

Gray line: $|\hat{C}^{(L_1)}(t) - \hat{C}^{(L_2)}(t)| / |\hat{C}^{(L_1)}(t)|$ for MPM (M2)

- MPM (M2) has similar performance compared to ESPRIT
- MPM (M1) has better control over the long tail
- MPM converges faster than AAA

Results — BCF (Ohmic bath)

$T = 0.001K$ $(\beta\omega_c \approx 7.19 \times 10^4)$



$J(\omega > 0) = \omega e^{-\omega/\omega_c}$ with $J(\omega < 0) = -J(-\omega)$

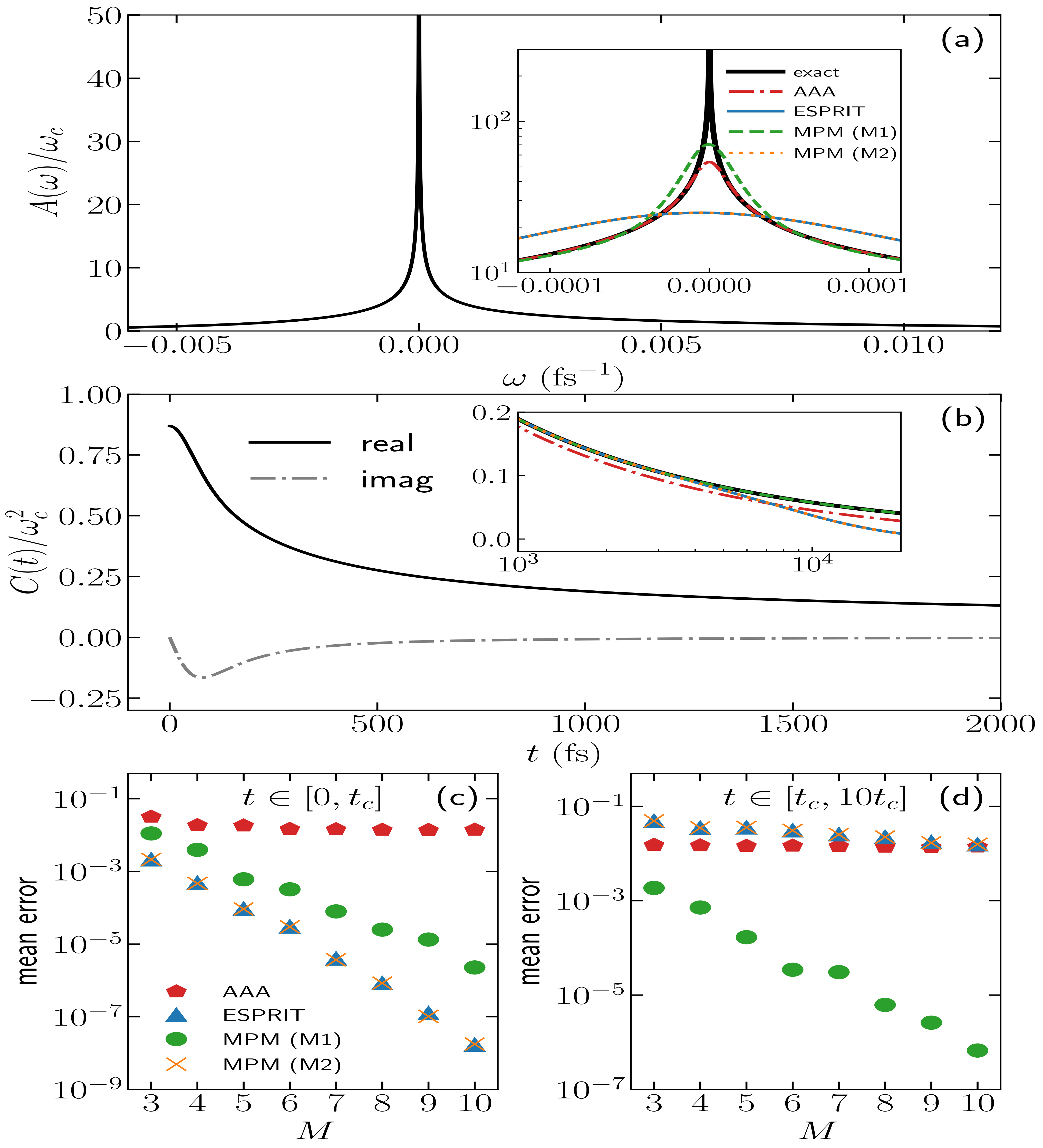
$$A(\omega) = J(\omega)\left[\coth\left(\frac{\beta\omega}{2}\right) + 1\right]$$

AAA: logarithmic discretized grid on $\omega \in [-\omega_{\max}, \omega_{\max}]$

ESPRIT: uniform grid on $t \in [0,\ t_c]$

Similar performance as for $T = 300K$

Results — BCF (Sub-Ohmic bath)



$J(\omega > 0) = (\omega_c \omega)^{0.5} e^{-\omega/\omega_c}$ with $J(\omega < 0) = -J(-\omega)$

$$A(\omega) = J(\omega)\left[\coth\left(\frac{\beta\omega}{2}\right) + 1\right]$$

AAA: logarithmic discretized grid on $\omega \in [-\omega_{\max}, \omega_{\max}]$

ESPRIT: uniform grid on $t \in [0,\ t_c]$

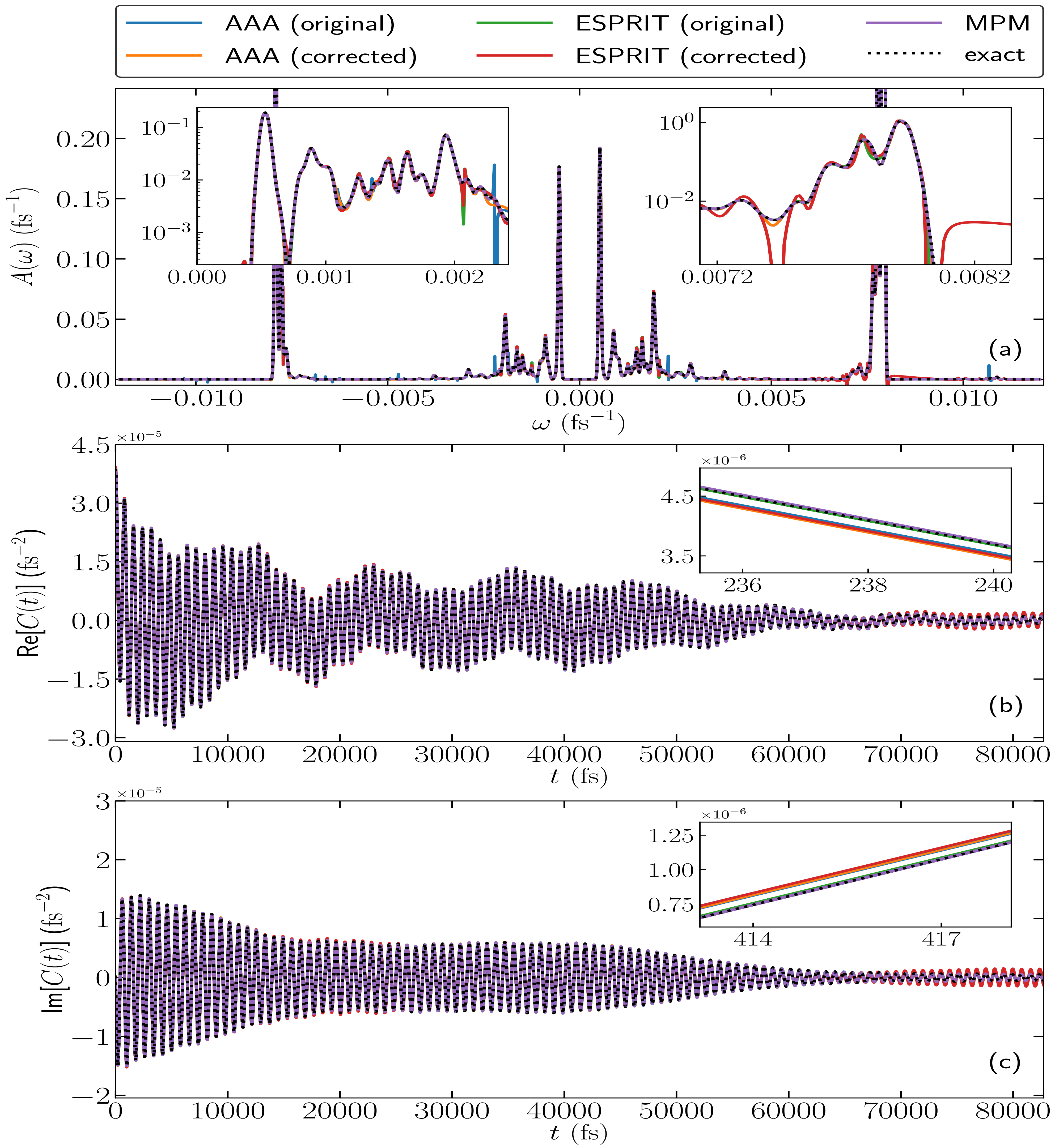
- AAA completely fails
- MPM (M2) has similar performance compared to ESPRIT
- MPM (M1) is the only one that has control over the long tail

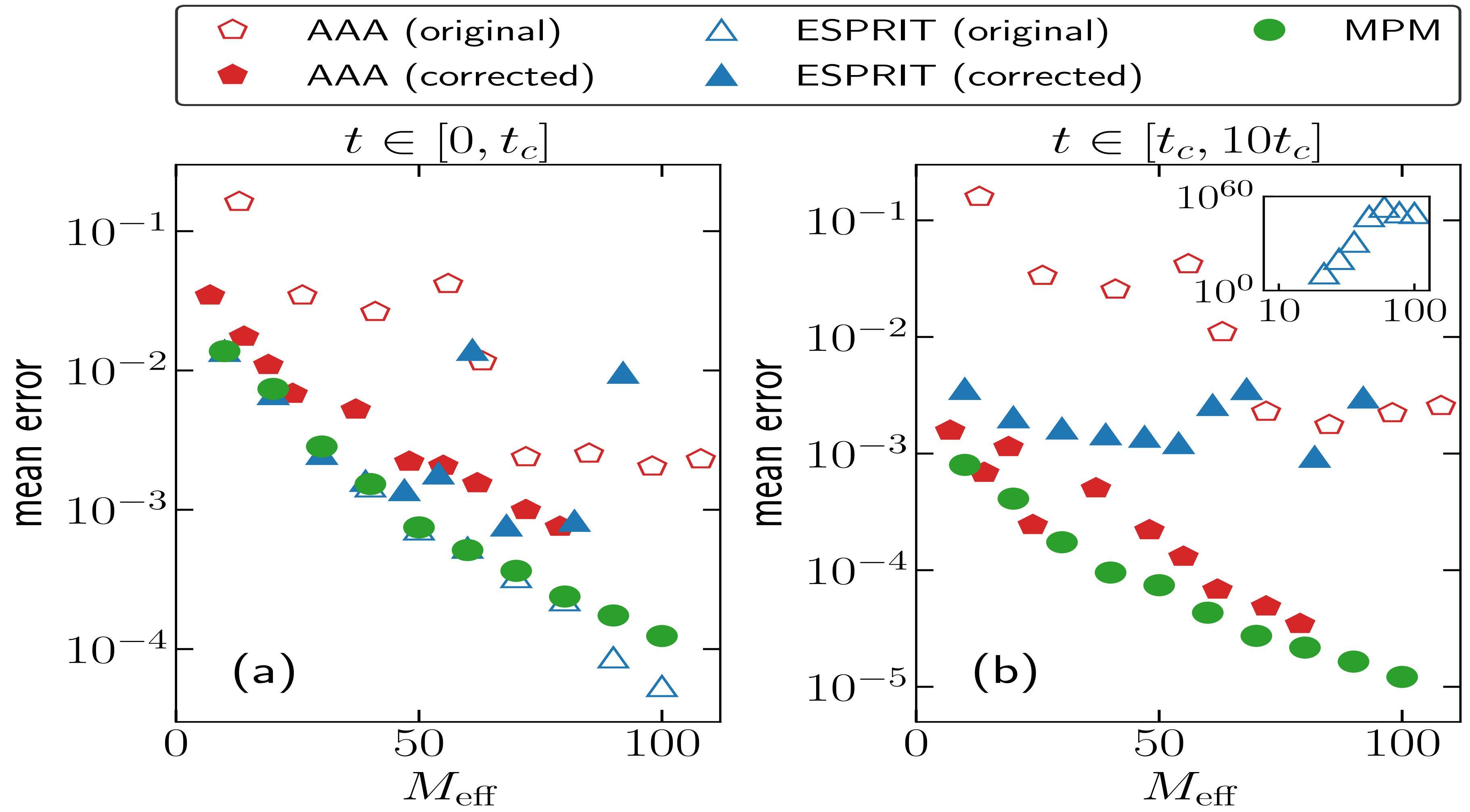
Reason: AAA approximates spectral shapes,
MPM approximates spectral moments

Results — BCF (Structured spectral function)

data from exciton–phonon couplings in quantum dots

Nano Lett. 21, 8741–8748 (2021)
npj Comput. Mater. 9, 145 (2023)





Input: $\{J(\omega_i), \omega_i\}$ on a uniform grid

Preprocessing: perform AAA with $\varepsilon = 10^{-8}$, leading to 704 poles, served as the exact solution

Issues: 1. ESPRIT recovers divergent exponentials

2. AAA recovers some real poles

Results: MPM remains stable and has better performance

Code

Thanks!

Green Software Package

https://github.com/Green-Phys/MiniPole

(also available: pypi, zenodo)

pip install mini_pole

MiniPoleRf: standard real-frequency MPM

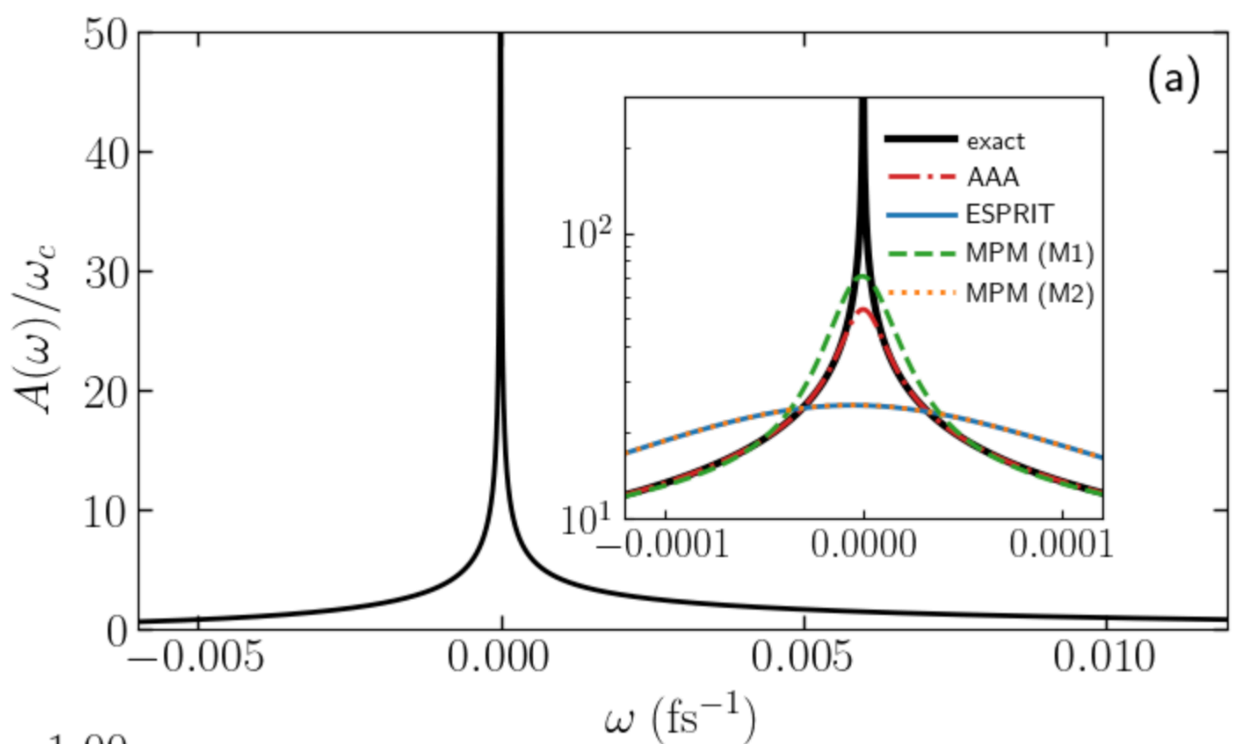
MiniPoleRfDPR: real-frequency MPM combined with AAA

MiniPole / examples / MPM_RF / subohmic_bath.ipynb

Preview | Code | Blame | 823 lines (823 loc) · 169 KB | Raw

```
ax21.semilogy(M_list, err_o_1, "^", color="tab:blue"  , markersize=markersize, label="ESPRIT")
ax21.semilogy(M_list, err_o_2, "o", color="tab:green" , markersize=markersize, label="MPM 1")
ax21.semilogy(M_list, err_o_3, "x", color="tab:orange", markersize=markersize, label="MPM 2")
ax21.tick_params(axis="both", which="both", direction="in", pad=3, labelsize=labelsize, right=True, top=True)
ax21.tick_params(which="major", length=major_len, width=major_wid)
ax21.tick_params(which="minor", length=minor_len, width=minor_wid)
ax21.set_xticks([3, 4, 5, 6, 7, 8, 9, 10])
ax21.set_yticks([1.e-7, 1.e-5, 1.e-3, 1.e-1])
ax21.yaxis.set_minor_locator(plt.NullLocator())
ax21.set_xlim([2.5, 10.5])
ax21.set_ylim([1.e-7, 5.e-1])
ax21.set_xlabel(r'$M$', fontsize=labelsize)
ax21.set_ylabel(r'mean error', fontsize=labelsize)
ax21.text(0.83, 0.9, '(d)', transform=ax21.transAxes, fontsize=fontsize)
ax21.text(0.5, 0.9, r'$t\in[t_c, 10t_c]$', transform=ax21.transAxes, fontsize=fontsize, ha="center")

plt.show()
```



If get stuck, feel free to ask me for help!

Lei Zhang

University of Michigan

lzphy@umich.edu