

# Faster QMC with Lagrange and Splines

W. D. Parker, J. W. Wilkins (Ohio State); R. G. Hennig, C. J. Umrigar (Cornell)

Support from DOE and NSF. Computing done at OSC, NERSC and NCSA.

## Wavefunction in QMC ( $\Psi$ )

- Frequent evaluation  $\langle E_L \rangle = \frac{1}{N_{MC}} \sum_{i=1}^{N_{MC}} \frac{\hat{H}\Psi}{\Psi} |_{\vec{R}=\vec{R}_i}$  statistical error  $\langle \sigma \rangle \propto \frac{1}{\sqrt{N_{MC}}}$
- Orbital basis representation: plane waves  
(single adjustable parameter, orthogonality)  $N_{PW} \propto N_{atoms}$
- Approximation methods: plane waves  $\mapsto$  polynomials |  $N_{poly} / \text{eval.} \ll N_{PW}$

Three types:

- Lagrange polynomial interpolation
- Piecewise-polynomial (pp-)spline interpolation
- Basis (B-)spline approximation

Features:

- Increase speed
- Keep plane wave accuracy
- Use more memory

**Three approximation methods equally accurate and faster  
B-splines win in memory use**

A.J. Williamson et al, Phys. Rev. Lett. **89**, 246406 (2001)

D. Alfè and M.J. Gillan, Phys. Rev. B **70**, 161101(R) (2004)

# Three Methods of Approximation - Lagrange, pp- & B-splines

## Lagrange polynomial interpolation

$$L(x, y, z) = \sum_{k=0}^3 \ell_k(z) \sum_{j=0}^3 \ell_j(y) \sum_{i=0}^3 \ell_i(x) \phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\vec{\nabla} L(x, y, z) = \sum_{k=0}^3 \ell_k(z) \sum_{j=0}^3 \ell_j(y) \sum_{i=0}^3 \ell_i(x) \vec{\nabla} \phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\nabla^2 L(x, y, z) = \sum_{k=0}^3 \ell_k(z) \sum_{j=0}^3 \ell_j(y) \sum_{i=0}^3 \ell_i(x) \nabla^2 \phi_{\text{PW}}(x_i, y_j, z_k)$$

## Piecewise-polynomial (pp-)spline interpolation

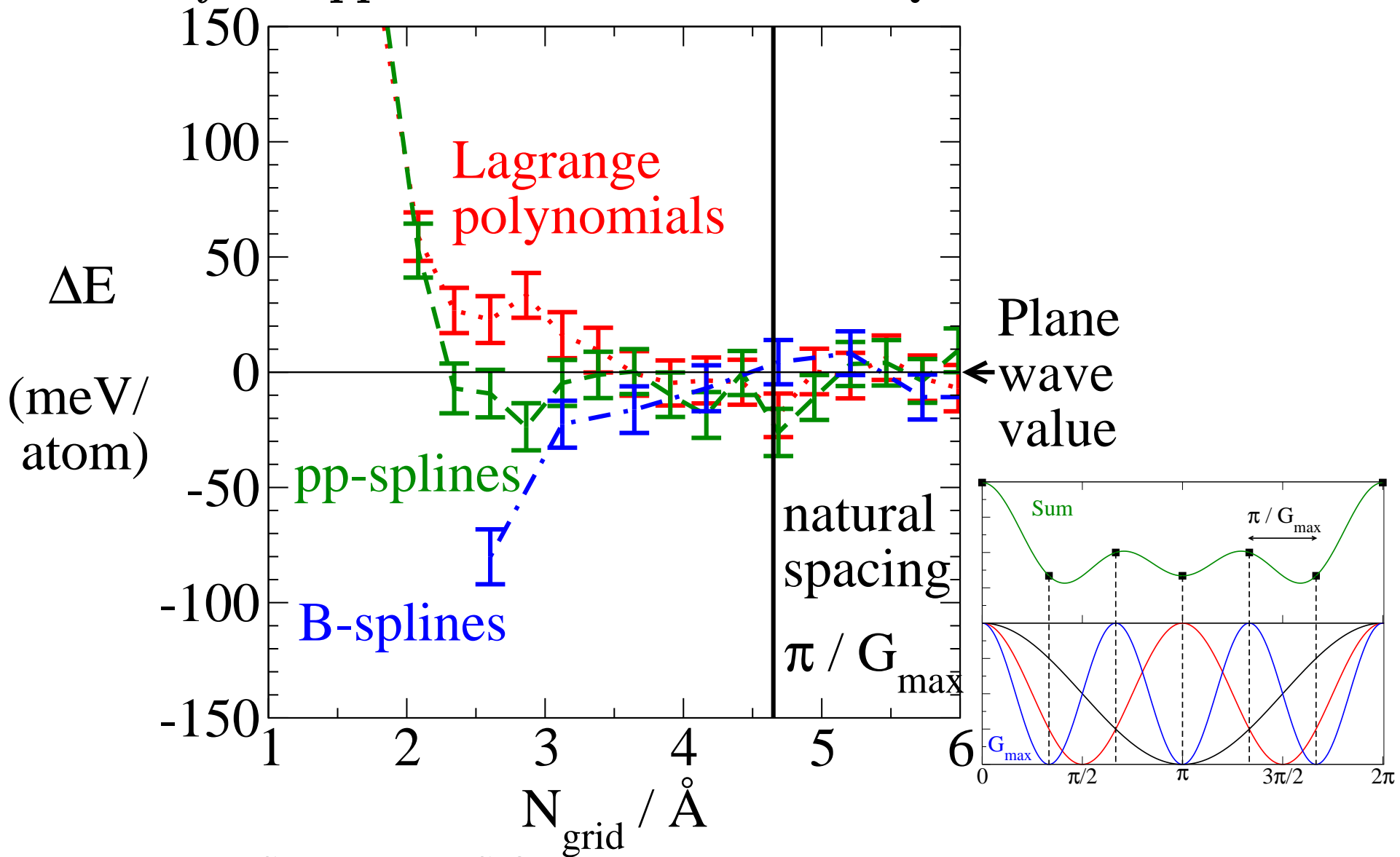
$$S(x, y, z) = \sum_{\mu=1}^8 \sum_{k=1}^2 s_k^\mu(z) \sum_{i=1}^2 s_i^\mu(x) \sum_{j=1}^2 s_j^\mu(y) \sigma_{ijk}^\mu, \quad \sigma_{ijk}^1 = \phi_{\text{PW}}(x_i, y_j, z_k)$$

## Basis (B-)spline approximation

$$B(x, y, z) = \sum_{i=0}^3 b_i(x) \sum_{j=0}^3 b_j(y) \sum_{k=0}^3 b_k(z) a_{ijk}, \quad a_{ijk} = \sum_{\nu=1}^{N_{\text{PW}}} \gamma_\nu \text{Re} \left( c_\nu \exp \left( i \vec{G}_\nu \cdot \vec{r}_{ijk} \right) \right)$$

All polynomials third degree. Cubic splines continuous up to second derivatives.

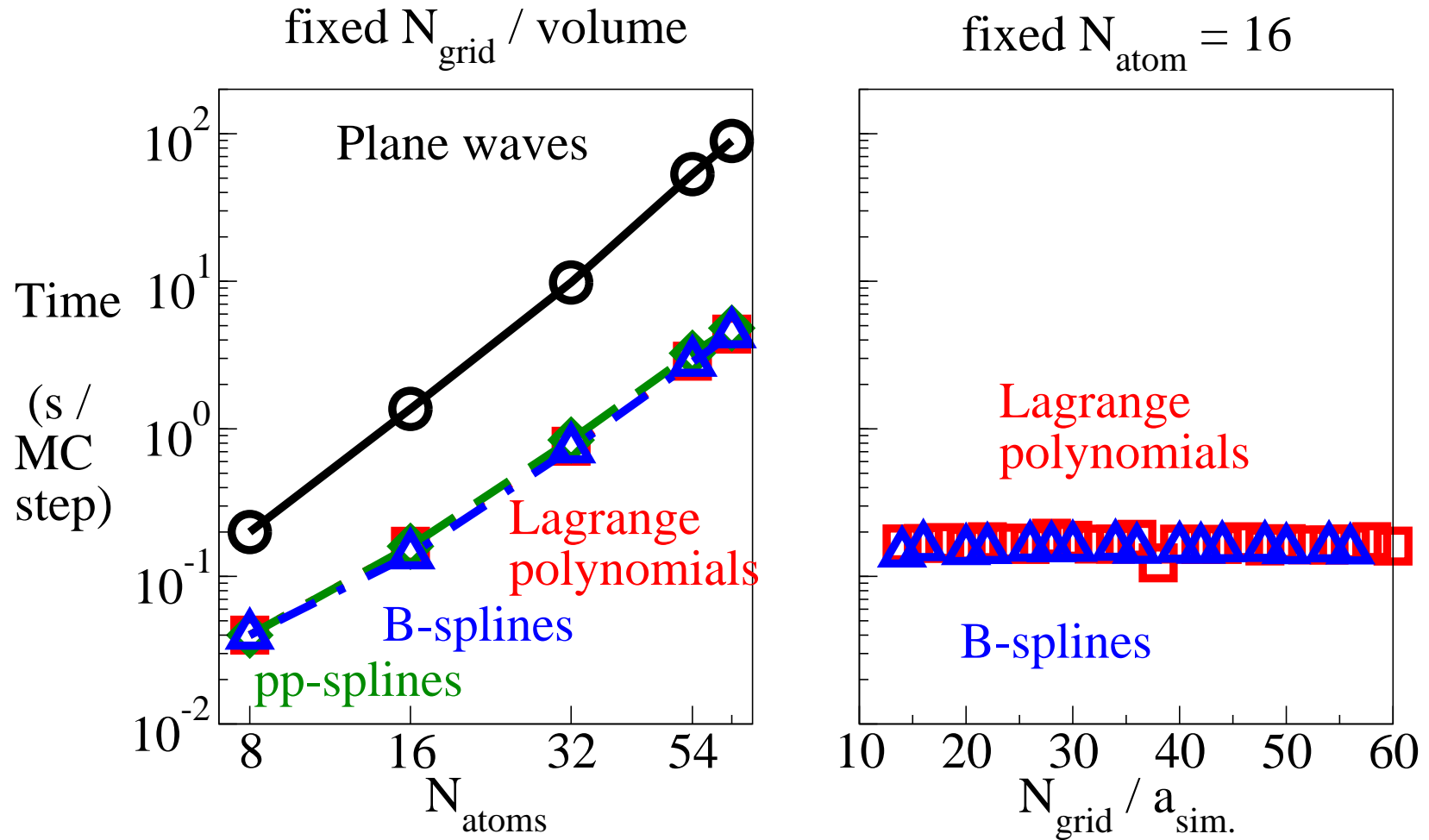
# Accuracy of Approximation Methods in QMC - Three Methods Good



Tested in: Si diamond, SiO<sub>2</sub> stishovite, bcc Ti

**Methods of comparable accuracy**

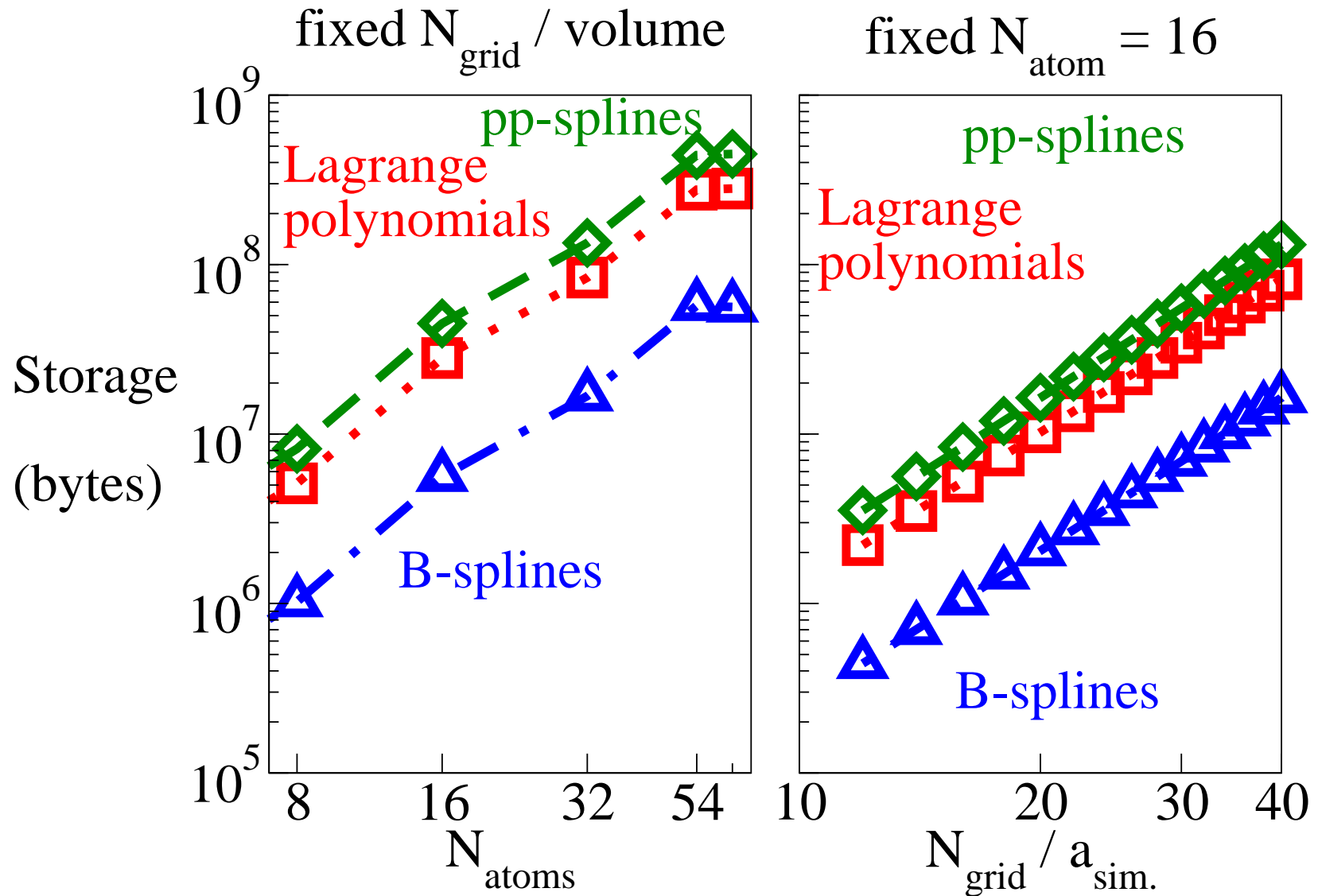
# Speed of Approximation Methods in QMC - Three Methods Good



Tested on: Itanium and Pentium platforms in serial and parallel

**Methods of comparable speed**

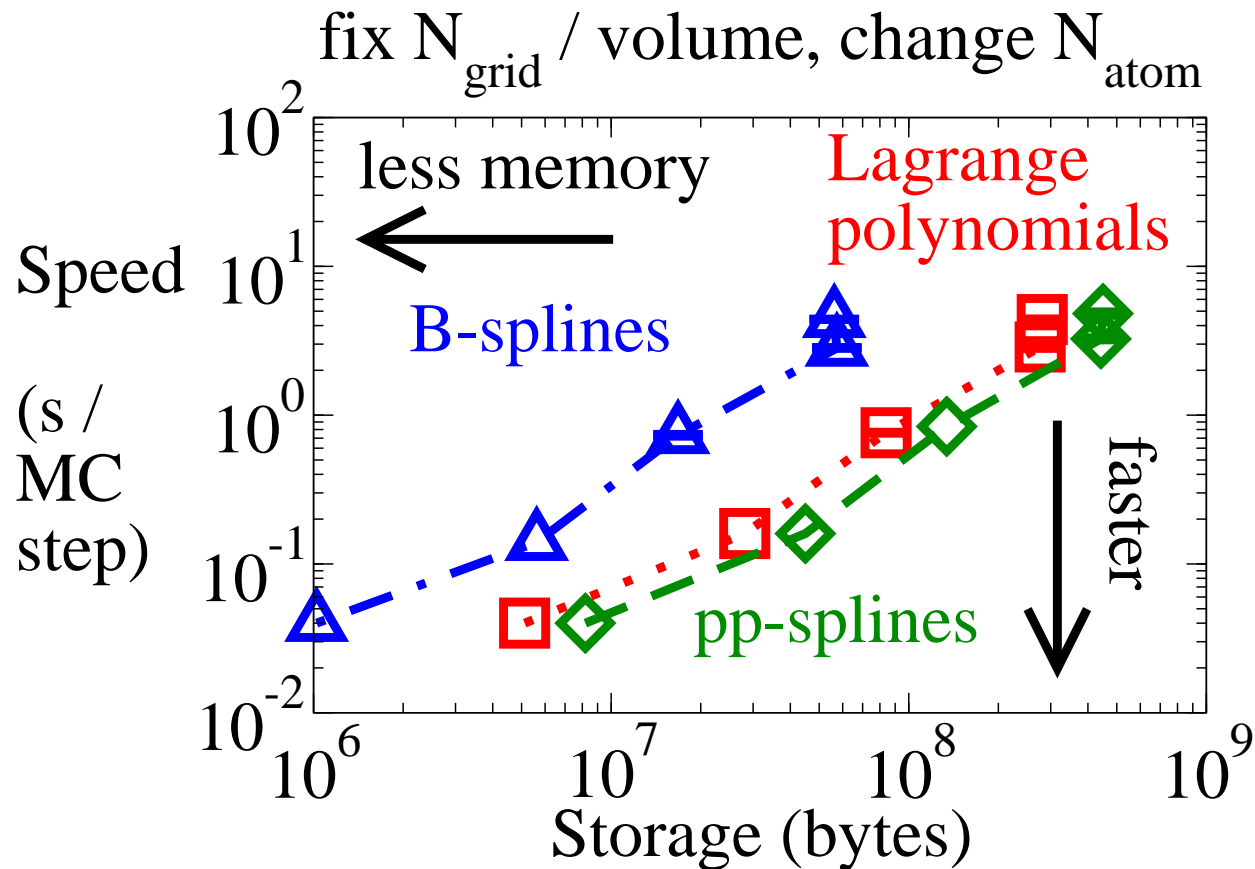
# Storage of Approximation Methods in QMC - B-splines Better



**B-splines require less storage (memory)**

# Conclusions

- All methods equally accurate to plane wave value
- All methods equally faster than plane waves
- B-splines win in memory ( $\frac{1}{8}$  pp-splines and  $\frac{1}{5}$  Lagrange)



**B-splines are better choice for faster QMC**

# Lagrange Polynomial Interpolation

- Derivatives possibly discontinuous  
 $\Rightarrow$  Interpolate gradient & Laplacian separately
- Store 5 values per grid point  $(\phi_{\text{PW}}, \vec{\nabla}\phi_{\text{PW}}, \nabla^2\phi_{\text{PW}})$

$$L(x, y, z) = \sum_{k=0}^3 \ell_k(\tilde{z}) \sum_{j=0}^3 \ell_j(\tilde{y}) \sum_{i=0}^3 \ell_i(\tilde{x}) \phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\vec{\nabla}L(x, y, z) = \sum_{k=0}^3 \ell_k(\tilde{z}) \sum_{j=0}^3 \ell_j(\tilde{y}) \sum_{i=0}^3 \ell_i(\tilde{x}) \vec{\nabla}\phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\nabla^2L(x, y, z) = \sum_{k=0}^3 \ell_k(\tilde{z}) \sum_{j=0}^3 \ell_j(\tilde{y}) \sum_{i=0}^3 \ell_i(\tilde{x}) \nabla^2\phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\tilde{x} = x - x_1 \quad 0 \leq \tilde{x} < 1 \quad \{x \leftrightarrow y \leftrightarrow z\}$$

$$\ell_0(\tilde{x}) = -\frac{1}{6}\tilde{x}(\tilde{x} - 1)(\tilde{x} - 2) \quad \ell_1(\tilde{x}) = \frac{1}{2}(\tilde{x} + 1)(\tilde{x} - 1)(\tilde{x} - 2)$$

$$\ell_2(\tilde{x}) = -\frac{1}{2}(\tilde{x} + 1)\tilde{x}(\tilde{x} - 2) \quad \ell_3(\tilde{x}) = \frac{1}{6}(\tilde{x} + 1)\tilde{x}(\tilde{x} - 1)$$

$$\tilde{x} \leftrightarrow \tilde{y} \leftrightarrow \tilde{z}$$

# Piecewise-Polynomial Spline Interpolation

- First and second derivatives continuous  
 $\Rightarrow$  Calculate gradient & Laplacian from spline
- Stores 8 points per grid point  $(\sigma^1, \sigma^2, \dots, \sigma^8)$

$$S(x, y, z) = \sum_{\mu=1}^8 \sum_{k=1}^2 s_k^\mu(\tilde{z}) \sum_{i=1}^2 s_i^\mu(\tilde{x}) \sum_{j=1}^2 s_j^\mu(\tilde{y}) \sigma_{ijk}^\mu, \quad \sigma_{ijk}^1 = \phi_{\text{PW}}(x_i, y_j, z_k)$$

$$\tilde{x} = x - x_1 \quad 0 \leq \tilde{x} < 1 \quad \{x \leftrightarrow y \leftrightarrow z\}$$

$$s_1^1(\tilde{x}) = s_1^3(\tilde{x}) = s_1^4(\tilde{x}) = s_1^6(\tilde{x}) = 1 - \tilde{x}$$

$$s_1^2(\tilde{x}) = s_1^5(\tilde{x}) = s_1^7(\tilde{x}) = s_1^8(\tilde{x}) = (1 - \tilde{x})((1 - \tilde{x})^2 - 1)$$

$$s_2^1(\tilde{x}) = s_2^3(\tilde{x}) = s_2^4(\tilde{x}) = s_2^6(\tilde{x}) = \tilde{x}$$

$$s_2^2(\tilde{x}) = s_2^5(\tilde{x}) = s_2^7(\tilde{x}) = s_2^8(\tilde{x}) = \tilde{x}(\tilde{x}^2 - 1)$$

$$s_1^1(\tilde{y}) = s_1^2(\tilde{y}) = s_1^4(\tilde{y}) = s_1^7(\tilde{y}) = 1 - \tilde{y}$$

$$s_1^3(\tilde{y}) = s_1^5(\tilde{y}) = s_1^6(\tilde{y}) = s_1^8(\tilde{y}) = (1 - \tilde{y})((1 - \tilde{y})^2 - 1)$$

$$s_2^1(\tilde{y}) = s_2^2(\tilde{y}) = s_2^4(\tilde{y}) = s_2^7(\tilde{y}) = \tilde{y}$$

$$s_2^3(\tilde{y}) = s_2^5(\tilde{y}) = s_2^6(\tilde{y}) = s_2^8(\tilde{y}) = \tilde{y}(\tilde{y}^2 - 1)$$

$$s_1^1(\tilde{z}) = s_1^2(\tilde{z}) = s_1^3(\tilde{z}) = s_1^5(\tilde{z}) = 1 - \tilde{z}$$

$$s_1^4(\tilde{z}) = s_1^6(\tilde{z}) = s_1^7(\tilde{z}) = s_1^8(\tilde{z}) = (1 - \tilde{z})((1 - \tilde{z})^2 - 1)$$

$$s_2^1(\tilde{z}) = s_2^2(\tilde{z}) = s_2^3(\tilde{z}) = s_2^5(\tilde{z}) = \tilde{z}$$

$$s_2^4(\tilde{z}) = s_2^6(\tilde{z}) = s_2^7(\tilde{z}) = s_2^8(\tilde{z}) = \tilde{z}(\tilde{z}^2 - 1)$$

# Basis Spline Approximation

- First and second derivatives continuous  
 $\Rightarrow$  Calculate gradient & Laplacian from spline
- Stores 1 points per grid point ( $a$ )
- Does not go through  $\phi_{\text{PW}}$  at grid points

$$B(x, y, z) = \sum_{i=0}^3 b_i(\tilde{x}) \sum_{j=0}^3 b_j(\tilde{y}) \sum_{k=0}^3 b_k(\tilde{z}) a_{ijk}$$

$$\tilde{x} = x - x_1 \quad 0 \leq \tilde{x} < 1 \quad \{x \leftrightarrow y \leftrightarrow z\}$$

$$b_0(\tilde{x}) = 2 - 3\tilde{x} + \frac{3}{2}\tilde{x}^2 - \frac{1}{4}\tilde{x}^3$$

$$b_1(\tilde{x}) = 1 - \frac{3}{2}\tilde{x}^2 + \frac{3}{4}\tilde{x}^3$$

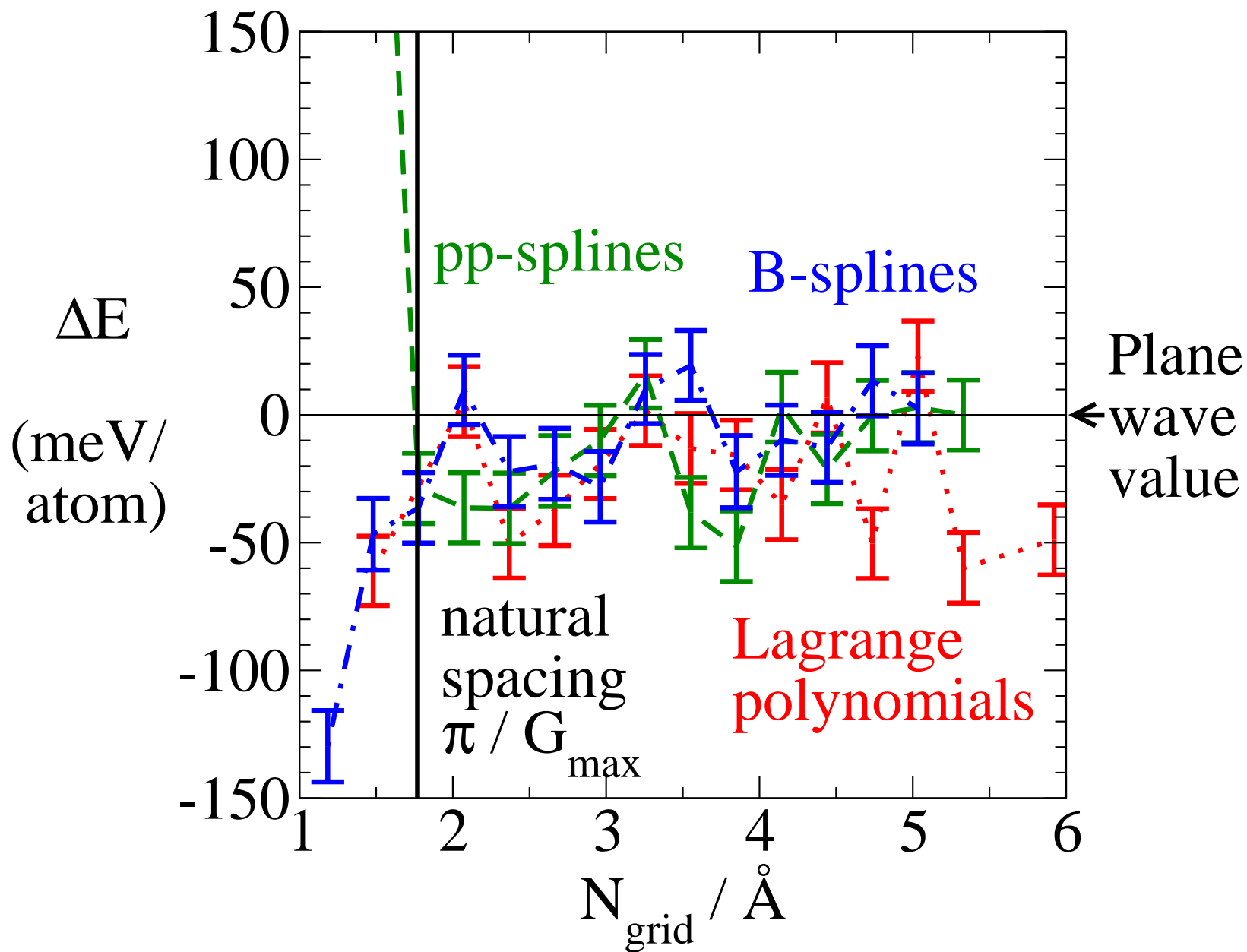
$$b_3(\tilde{x}) = 1 - \frac{3}{2}\tilde{x}^2 - \frac{3}{4}\tilde{x}^3$$

$$b_4(\tilde{x}) = 2 + 3\tilde{x} + \frac{3}{2}\tilde{x}^2 + \frac{1}{4}\tilde{x}^3$$

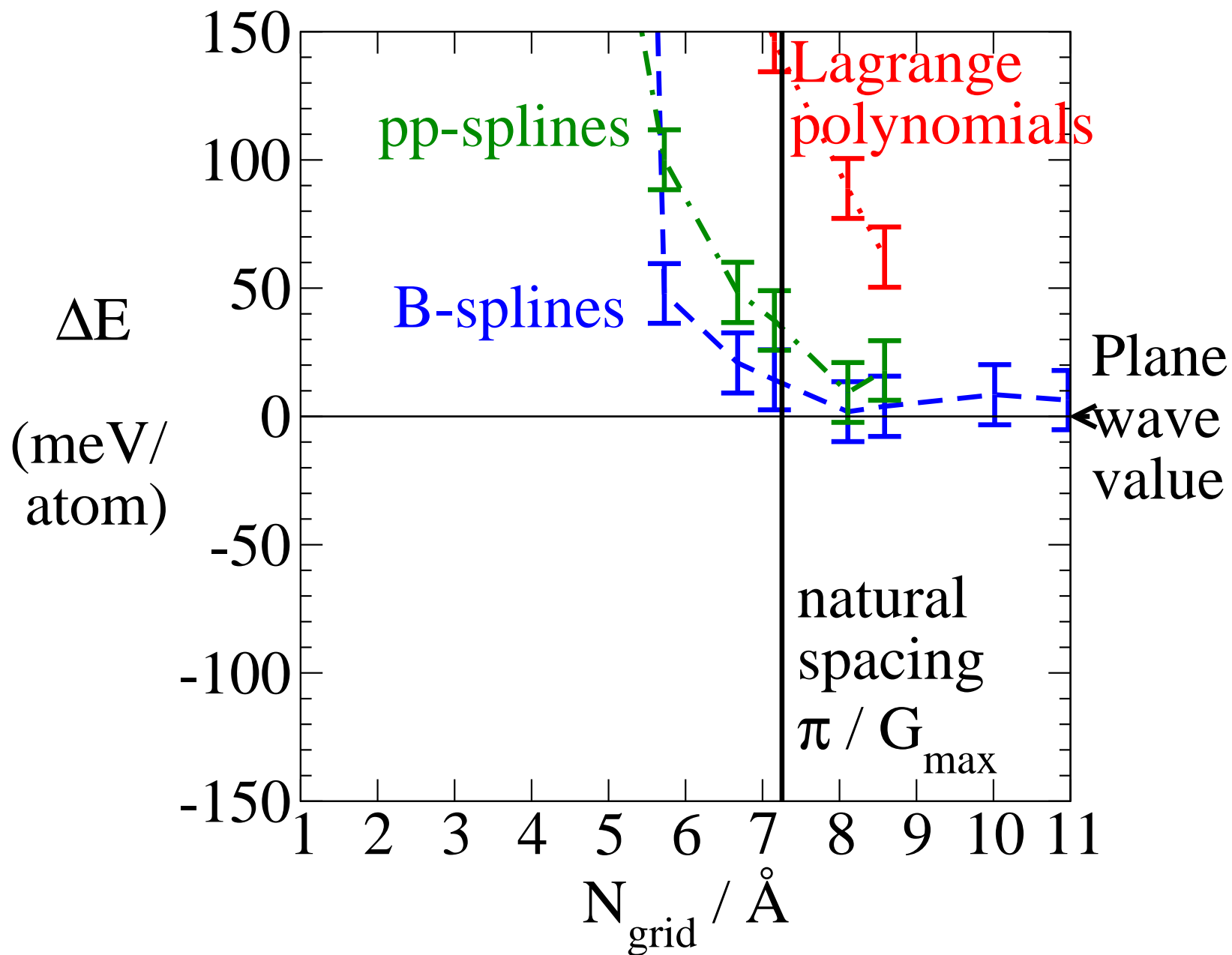
$$a_{ijk} = \sum_{\nu=1}^{N_{\text{PW}}} \gamma_{\nu} \text{Re} \left( c_{\nu} \exp \left( i \vec{G}_{\nu} \cdot \vec{r}_{ijk} \right) \right)$$

$$\gamma_{\nu} = \prod_{w \in x, y, z} \frac{3}{G_{\nu, w}} (3 - 4 \cos(G_{\nu, w}) + \cos(2G_{\nu, w}))$$

# Accuracy of Approximation Methods in QMC - bcc Ti

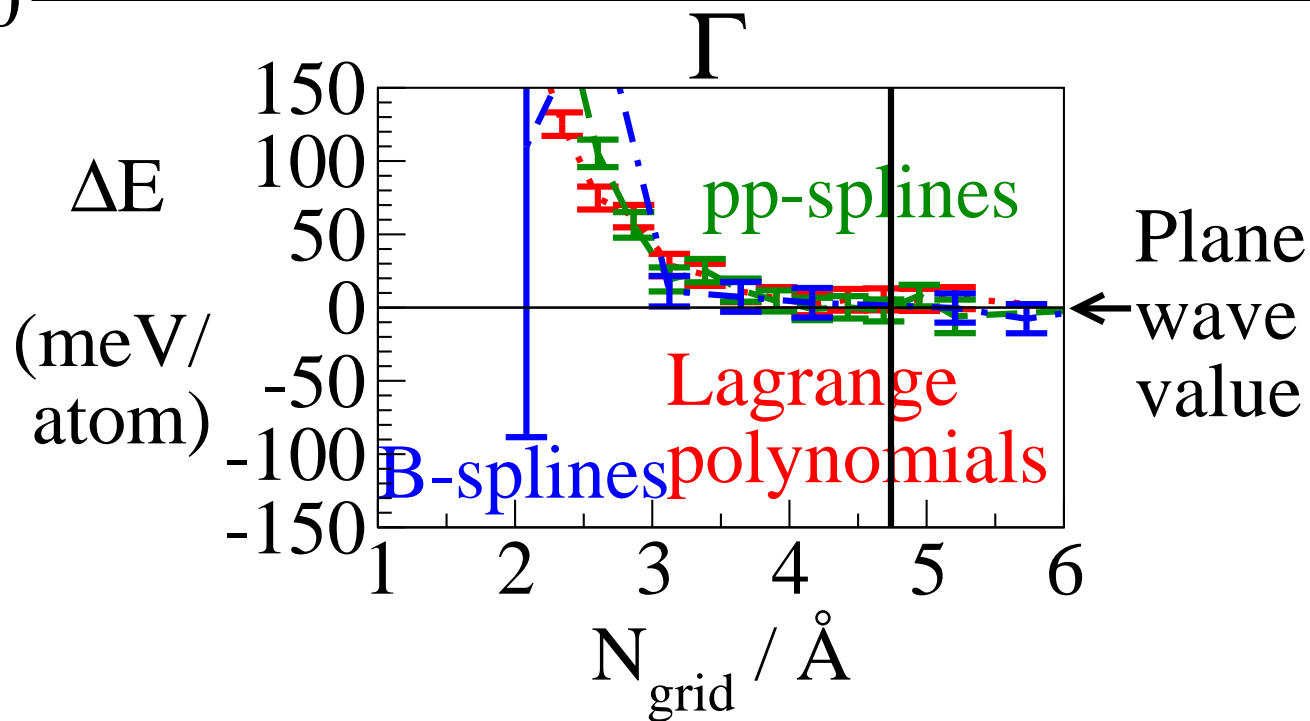
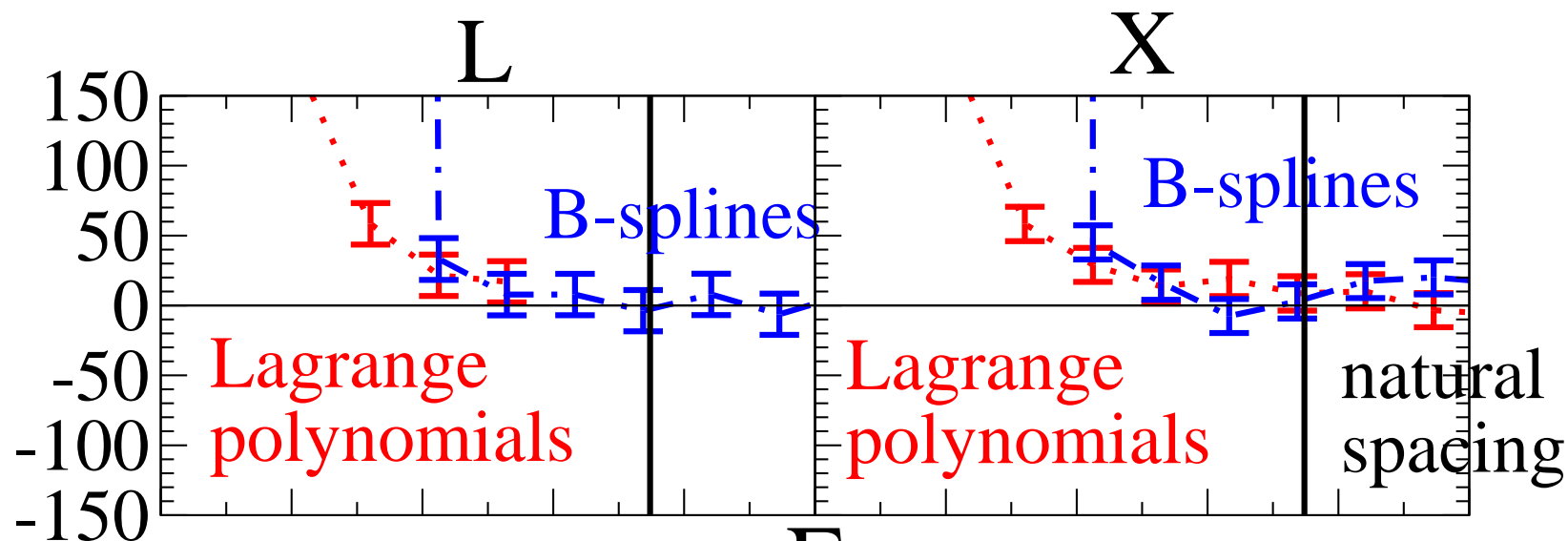


# Accuracy of Approximation Methods in QMC - stishovite SiO<sub>2</sub>



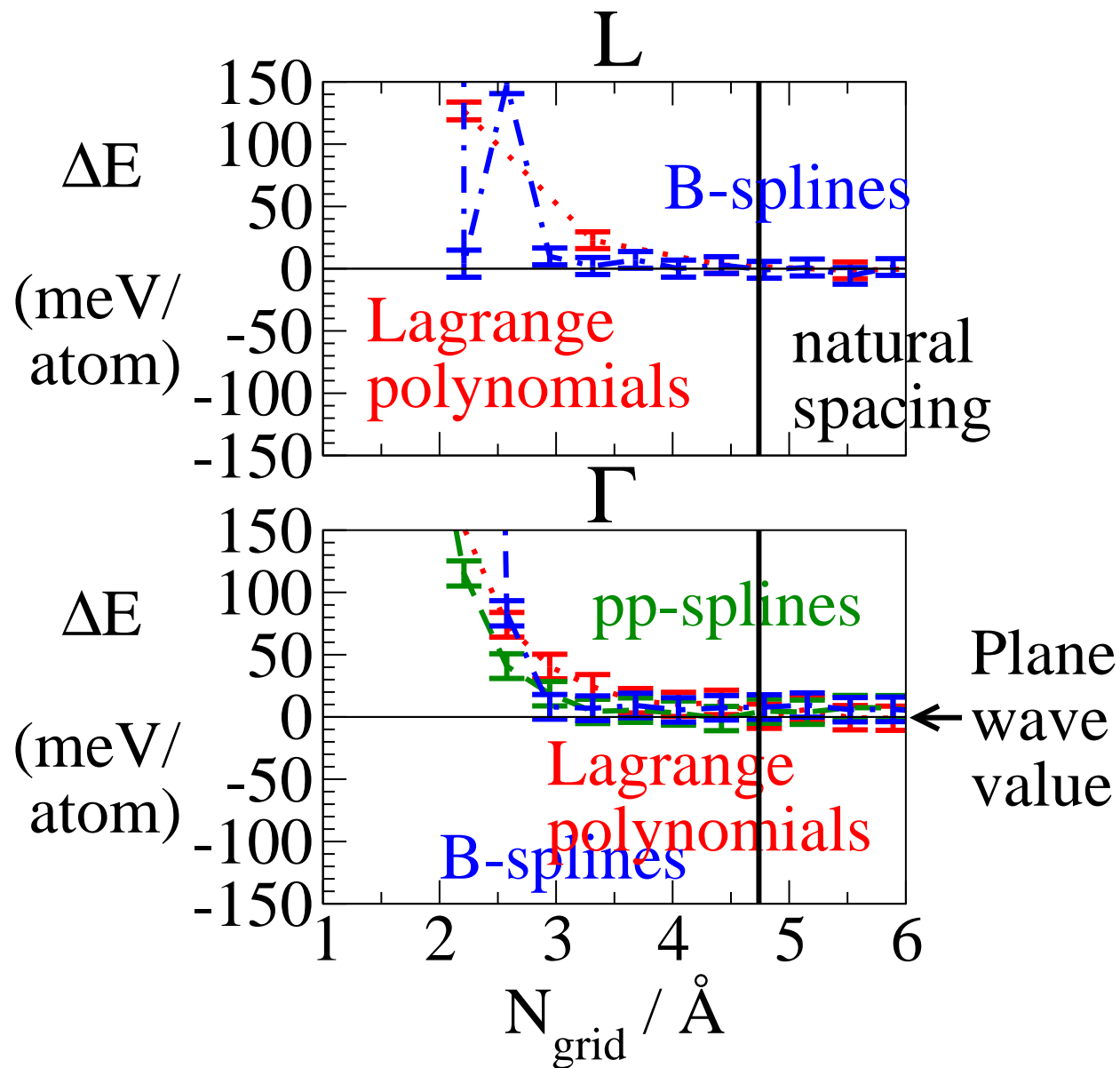
# Accuracy of Approximation Methods in QMC - diamond Si k-points

$$N_{\text{atom}} = 2$$



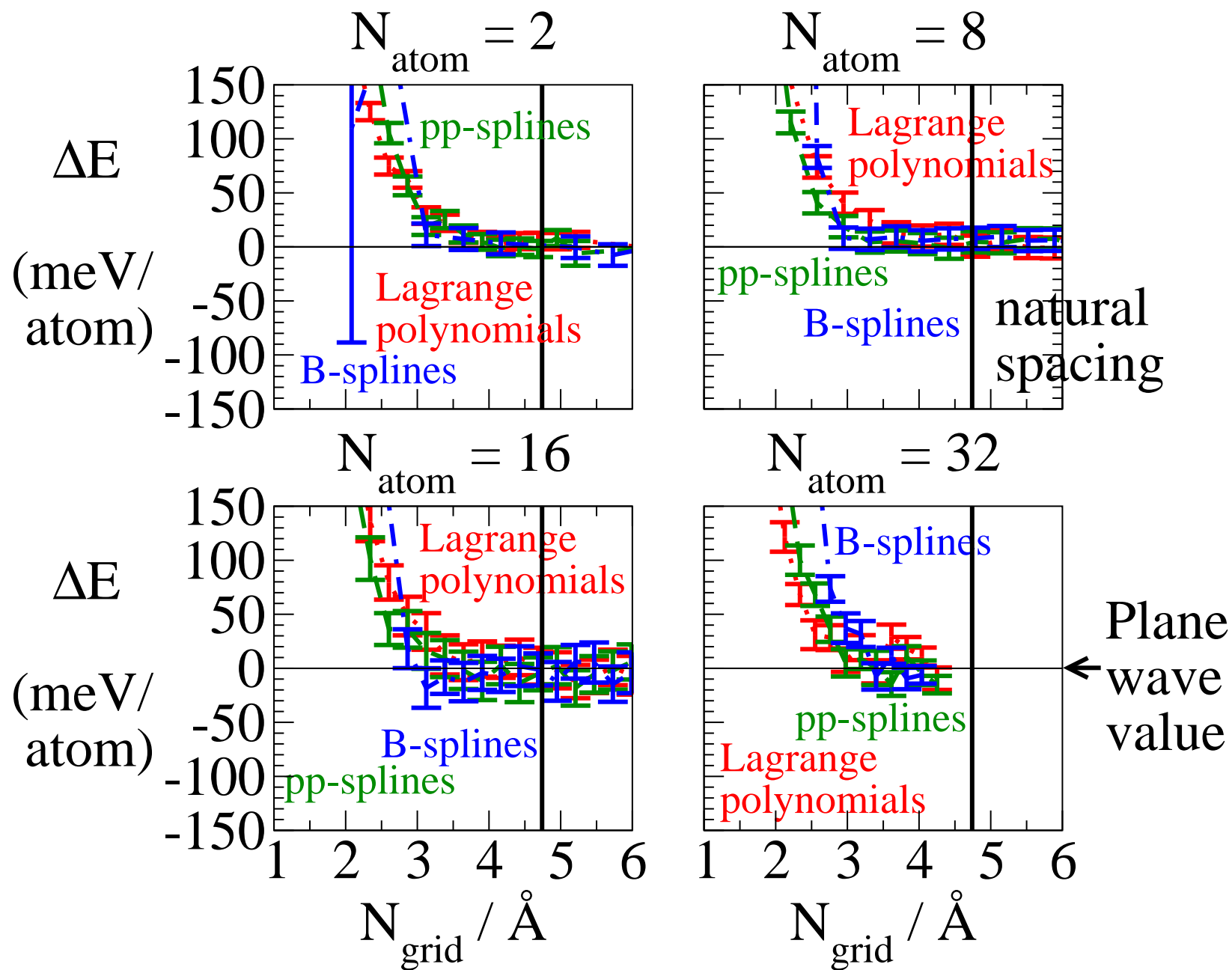
# Accuracy of Approximation Methods in QMC - diamond Si k-points

$$N_{\text{atom}} = 8$$



# Accuracy of Approximation Methods in QMC - diamond Si supercells

## VMC



# Accuracy of Approximation Methods in QMC - diamond Si supercells

## DMC

