## **Numerical operator calculus in higher dimensions**

## **Gregory Beylkin\* and Martin J. Mohlenkamp**

Applied Mathematics, University of Colorado, Boulder, CO 80309

Communicated by Ronald R. Coifman, Yale University, New Haven, CT, May 31, 2002 (received for review July 31, 2001)

**When an algorithm in dimension one is extended to dimension** *d***, in nearly every case its computational cost is taken to the power** *d***. This fundamental difficulty is the single greatest impediment to solving many important problems and has been dubbed the** *curse of dimensionality***. For numerical analysis in dimension** *d***, we propose to use a representation for vectors and matrices that generalizes separation of variables while allowing controlled accuracy. Basic linear algebra operations can be performed in this representation using one-dimensional operations, thus bypassing the exponential scaling with respect to the dimension. Although not all operators and algorithms may be compatible with this representation, we believe that many of the most important ones** are. We prove that the multiparticle Schrödinger operator, as well **as the inverse Laplacian, can be represented very efficiently in this form. We give numerical evidence to support the conjecture that eigenfunctions inherit this property by computing the ground-**

state eigenfunction for a simplified Schr**öblingeichingfalmmangildnis0pfoldinibfigppinnt顯(的表現)>5dli(am);Gdftbinalple)-548(of)-548(the)-548(ex)-30(ponential)-548 [[(th;)/adib(sm):1ath(s)/divid/and/butch-348(af)-548(the)-548(ex)-30(ponential)-548** 

 $\phi$  $\left( \begin{array}{c} 0 \end{array} \right)$  nor try to find and solve equations for  $\phi$ *l* (*x* )}. We  $\mathbf{2}$  since  $\mathbf{2}$  si

This paper was submitted directly (Track II) to the PNAS office. Abbreviation: SVD, singular value decomposition.



**The Multiparticle Schro¨ dinger Operator**





**Basic Linear Algebra**



This linear least-squares problem naturally divides into separate problems for each coordinate. For fixed direction *k*, form the matrix with entries

$$
B( , \cdot ) = \tilde{\mathbf{V}}, \tilde{\mathbf{V}}.
$$
 [22]

 $\mathbf{b}$ , for a fixed coordinate  $\mathbf{b}$ 

$$
V(\mathbf{v}) = \sum_{\mathbf{v} \in \mathcal{V}} V(\mathbf{v}, \mathbf{v}) \mathbf{v} \mathbf{v} \mathbf{v} \tag{23}
$$

The normal equations for the direction *k* and coordinate *<sup>k</sup>*

$$
\mathbb{B} \quad (\cdot) = \mathbf{b} \ , \tag{24}
$$

which we solve for *c <sup>k</sup>* (*l ˜*) as a vector in *l ˜*. After computing *<sup>c</sup> <sup>k</sup>* (*l ˜*) for all coordinates *<sup>k</sup>*, we let *˜sl ˜ c <sup>k</sup>* (*l ˜*) and *V˜ <sup>k</sup> ˜* ( *<sup>k</sup>*) *c <sup>k</sup>* (*l ˜*) *˜sl ˜*, where the norm is taken with respect to the coordinate *<sup>k</sup>*. For fixed direction *k* and coordinate *<sup>k</sup>*, it requires *˜r*2*dM* operations to compute , *˜rrdM* to compute **b** *<sup>k</sup>* , and *˜r*<sup>3</sup> to solve the system. Since and the inner products in **b** *<sup>k</sup>* **<sup>b</sup>**



