Supplementary information was updated on the 7th Feburary 2025 to correct equation S80

Electronic Supplementary Information for

Introducing DDEC6 atomic population analysis: Part 4. Efficient parallel computation of net atomic charges, atomic spin moments, bond orders, and more

Nidia Gabaldon Limas and Thomas A. Manz*
Department of Chemical & Materials Engineering, New Mexico State University, Las Cruces, New Mexico, 88003-8001.

*email: tmanz@nmsu.edu

Note: This revised ESI pdf corrected a typo in eqn S80 that was present in the originally published ESI pdf. The authors apologize for any inconvenience this may have caused to readers.

CONTENTS:

S1. The 14 charge partitioning Lagrangians

The seven stockholder Lagrangians have the form

$$F^{(i)} = \sum_A \oint \rho_A^{(i)}(\vec{r}_A)\left[\ln\left(\frac{\rho_A^{(i)}(\vec{r}_A)}{H_A^{(i)}(\vec{r}_A)}\right)-1\right]d^3\vec{r}_A + \int_U \gamma^{(i)}(\vec{r})\Theta^{(i)}(\vec{r})d^3\vec{r} - \sum_A \kappa_A^{(i)}\Psi_A^{(i)} \qquad \text{(S1)}$$

(To include enough distinct letters for mathematical symbols, we used characters from the Roman, Greek, and Cyrillic alphabets. In this article, i and j are used to represent many different kinds of indices.) This stockholder Lagrangian form is uniquely derivable from the condition that the ratio $\frac{\left(\rho_A^{(i)}(\vec{r}_A)/H_A^{(i)}(\vec{r}_A)\right)}{\left(\rho_B^{(i)}(\vec{r}_B)/H_B^{(i)}(\vec{r}_B)\right)}$ should be independent of the position $\vec{r}$, where $H_A^{(i)}(\vec{r}_A)$ is some fixed target atomic pseudo-density distribution and $\rho_A^{(i)}(\vec{r}_A)$ is the assigned atomic electron density distribution subject to constraints (S2) and (S4) below. Eq. (S1) is the only possible stockholder Lagrangian form satisfying this condition.

(*Proof*: The variational derivative, $\frac{\delta}{\delta\rho_A(\vec{r}_A)}$, of constraints (S2) and (S4) yield $-\gamma^{(i)}(\vec{r})-\kappa_A^{(i)}$. This necessarily produces $\frac{\left(\rho_A^{(i)}(\vec{r}_A)/H_A^{(i)}(\vec{r}_A)\right)}{\left(\rho_B^{(i)}(\vec{r}_B)/H_B^{(i)}(\vec{r}_B)\right)}$ independent of position $\vec{r}$ only if $\frac{\delta}{\delta\rho_A(\vec{r}_A)}$ of the unconstrained part of the Lagrangian is $\ln\left(\frac{\rho_A^{(i)}(\vec{r}_A)}{H_A^{(i)}(\vec{r}_A)}\right)$ (optionally times or plus constants that do not change the optimized $\left\{\rho_A^{(i)}(\vec{r}_A)\right\}$). Setting $\frac{\delta F^{(i)}}{\delta\rho_A(\vec{r}_A)}=\ln\left(\frac{\rho_A^{(i)}(\vec{r}_A)}{H_A^{(i)}(\vec{r}_A)}\right)-\gamma^{(i)}(\vec{r})-\kappa_A^{(i)}$ equal to zero yields the solution, $\frac{\rho_A^{(i)}(\vec{r}_A)}{H_A^{(i)}(\vec{r}_A)}=e^{\kappa_A^{(i)}}e^{\gamma^{(i)}(\vec{r})}$, which gives $\frac{\left(\rho_A^{(i)}(\vec{r}_A)/H_A^{(i)}(\vec{r}_A)\right)}{\left(\rho_B^{(i)}(\vec{r}_B)/H_B^{(i)}(\vec{r}_B)\right)}=e^{\kappa_A^{(i)}-\kappa_B^{(i)}}$ independent of position $\vec{r}$.) It is a special case of Nalewajski and Parr's stockholder Lagrangian form.[S1]

Here, $\gamma^{(i)}(\vec{r})$ is a Lagrange multiplier that enforces the constraint

$$\Theta^{(i)}(\vec{r})=\rho(\vec{r})-\sum_{\ell,B}\rho_B^{(i)}(\vec{r}_B)\Rightarrow 0 \tag{S2}$$

where

$$\sum_{\ell,B}=\sum_{\ell_1}\sum_{\ell_2}\sum_{\ell_3}\sum_{B} \tag{S3}$$

$\kappa_A^{(i)}$ is a Lagrange multiplier that enforces the constraint

$$\text{Ч}_A^{(i)}\geq 0 \tag{S4}$$

These $\kappa_A^{(i)}$ constraints are nil for the first four charge partitioning steps:

$$\text{Ч}_A^{(1,2,3,4)}=0 \tag{S5}$$

$$\kappa_A^{(1,2,3,4)}=0 \tag{S6}$$

They are potent for the last three charge partitioning steps:

$$\text{Ч}_A^{(5,6,7)}=N_A^{(5,6,7)}-N_A^{\text{core}} \tag{S7}$$

$$\kappa_A^{(5,6,7)}\geq 0 \tag{S8}$$

$$N_A^{(i)}=\oint\rho_A^{(i)}(\vec{r}_A)\,d^3\vec{r}_A \tag{S9}$$

$$q_A^{(i)}=z_A-N_A^{(i)} \tag{S10}$$

where z_A is the nuclear charge, N_A^{core} is the number of core electrons, N_A is the number of electrons, and q_A is the NAC, all assigned to atom A.

The minimum is obtained by setting the variational derivative to zero

$$\delta F^{(i)}\Rightarrow 0 \tag{S11}$$

This minimum is unique, because the curvature is positive definite:

$$\delta^2F^{(i)}=\sum_A\oint\left(\left(\delta\rho_A^{(i)}(\vec{r}_A)\right)^2/\rho_A^{(i)}(\vec{r}_A)\right)d^3\vec{r}_A>0 \tag{S12}$$

The solution to this minimization problem is

$$w_A^{(i)}\left(\vec{r}_A\right) = H_A^{(i)}\left(\vec{r}_A\right) e^{\kappa_A^{(i)}} \tag{S13}$$

$$W^{(i)}\left(\vec{r}\right) = \sum_{\ell,B} w_B^{(i)}\left(\vec{r}_B\right) = \rho\left(\vec{r}\right) \exp\left(-\gamma^{(i)}\left(\vec{r}\right)\right) \tag{S14}$$

$$\rho_A^{(i)}\left(\vec{r}_A\right) = w_A^{(i)}\left(\vec{r}_A\right) \rho\left(\vec{r}\right) / W^{(i)}\left(\vec{r}\right) \geq 0 \tag{S15}$$

The final DDEC6 charge partitions are defined by

$$\rho_A\left(\vec{r}_A\right) = \rho_A^{(7)}\left(\vec{r}_A\right) \tag{S16}$$

All that remains is to define $H_A^{(i)}\left(\vec{r}_A\right)$ for each of the seven charge partitioning steps. The 1st charge partitioning step is defined by

$$H_A^{(1)}\left(\vec{r}_A\right) = \frac{\overline{\rho}_A^{ref}\left(r_A, 0\right)}{3\sum_{\ell,B} \overline{\rho}_B^{ref}\left(r_B, 0\right)} + \frac{2\left(\overline{\rho}_A^{ref}\left(r_A, 0\right)\right)^4}{3\sum_{\ell,B}\left(\overline{\rho}_B^{ref}\left(r_B, 0\right)\right)^4} \tag{S17}$$

The 2nd charge partitioning step is defined by

$$H_A^{(2)}\left(\vec{r}_A\right) = \frac{\overline{\rho}_A^{ref}\left(r_A, q_A^{(1)}\right)}{3\sum_{\ell,B} \overline{\rho}_B^{ref}\left(r_B, q_B^{(1)}\right)} + \frac{2\left(\overline{\rho}_A^{ref}\left(r_A, q_A^{(1)}\right)\right)^4}{3\sum_{\ell,B}\left(\overline{\rho}_B^{ref}\left(r_B, q_B^{(1)}\right)\right)^4} \tag{S18}$$

Both the 1st and 2nd charge partitioning steps use a combination of stockholder and localized charge partitioning to ensure the computed reference ion charge is similar to the number of electrons in the volume dominated by that atom.

The 3rd charge partitioning step uses:

$$H_A^{(3)}\left(\vec{r}_A\right) = \overline{\rho}_A^{ref}\left(r_A, q_A^{(2)}\right) \tag{S19}$$

During the third charge partitioning step, the 4th Lagrangian is used to compute the conditioned reference ion densities, $\left\{\rho_A^{cond}\left(r_A\right)\right\}$, via reshaping that enforces the constraints

$$\phi^I\left(r_A\right) = \frac{d\rho_A^{cond}\left(r_A\right)}{dr_A} \leq 0 \tag{S20}$$

and to integrate to the number of electrons in the reference ion:

$$\varphi_A^I = \int_0^{r_{cutoff}} \rho_A^{cond}\left(r_A\right) 4\pi\left(r_A\right)^2 dr_A - z_A + q_A^{(2)} \Rightarrow 0 \tag{S21}$$

These constraints were introduced for theoretical appeal to ensure expected behavior. In tests we performed, these constraints had only a small effect on the NACs. They are not present in the DDEC3 method. $\left\{\rho_A^{cond}\left(r_A\right)\right\}$ is found by minimizing the functional

$$h^I\left(\rho_A^{cond}\left(r_A\right)\right) = \int_0^{r_{cutoff}} \left[\frac{\left(\rho_A^{cond}\left(r_A\right) - \rho_A^{(3),avg}\left(r_A\right)\right)^2}{2\sqrt{\rho_A^{(3),avg}\left(r_A\right)}} + \Gamma_A^I\left(r_A\right)\phi^I\left(r_A\right)\right] 4\pi\left(r_A\right)^2 dr_A - \Phi_A^I \varphi_A^I \tag{S22}$$

where $\Gamma_A^I(r_A)$ and Φ_A^I are Lagrange multipliers enforcing constraints (S20) and (S21), respectively. $h^I\left(\rho_A^{cond}(r_A)\right)$ is a convex functional with the unique minimum:

$$\rho_A^{cond}(r_A) = \rho_A^{(3),avg}(r_A) + \sqrt{\rho_A^{(3),avg}(r_A)}\left(\Phi_A^I + \frac{d\Gamma_A^I(r_A)}{dr_A} + \frac{2\Gamma_A^I(r_A)}{r_A}\right) \tag{S23}$$

The fourth charge partitioning step uses

$$H_A^{(4)}(\vec{r}_A) = \rho_A^{cond}(r_A) \tag{S24}$$

to construct the 5th Lagrangian. During the 4th, 5th, and 6th charge partitioning steps, the following weighted spherical average is computed:

$$\rho_A^{(4,5,6),wavg}(r_A) = \frac{\theta^{(4,5,6)}(r_A) + \frac{\rho_A^{(4,5,6),avg}(r_A)}{5}\left\langle \frac{w_A^{(4,5,6)}(r_A)}{W^{(4,5,6)}(\vec{r})}\right\rangle_{r_A}}{1 - \frac{4}{5}\left\langle \frac{w_A^{(4,5,6)}(r_A)}{W^{(4,5,6)}(\vec{r})}\right\rangle_{r_A}} \tag{S25}$$

$$\theta^{(4,5,6)}(r_A) = \left\langle \left(1 - \frac{w_A^{(4,5,6)}(r_A)}{W^{(4,5,6)}(\vec{r})}\right)\rho_A^{(4,5,6)}(\vec{r}_A)\right\rangle_{r_A} \tag{S26}$$

Using this weighted spherical average improves the accuracy of charge partitioning compared to using a simple spherical average, because it weighs more heavily those regions of space where atoms overlap in order to reduce the atomic dipoles.[S2]

During the 4th, 5th, and 6th charge partitioning steps, the 6th, 9th, and 12th Lagrangians are used to perform reshaping to ensure the tails of buried atoms do not become too diffuse. This is done by enforcing the constraints

$$\phi^{II}(r_A) = \frac{dG_A(r_A)}{dr_A} + \eta_A^{lower}(r_A)G_A(r_A) \le 0 \tag{S27}$$

$$\varphi_A^{II} = \int_0^{r_{cutoff}} \left(G_A(r_A) - \rho_A^{wavg}(r_A)\right)4\pi(r_A)^2\,dr_A \Rightarrow 0 \tag{S28}$$

where

$$\eta_A^{lower}(r_A) = \left(1.75\,\text{bohr}^{-1}\right)\left(1 - \left(\tau_A(r_A)\right)^2\right) \tag{S29}$$

$$\tau_A(r_A) = \left\langle \frac{\rho_A^{cond}(r_A)}{\sqrt{\rho^{cond}(\vec{r})}}\right\rangle_{r_A}\left(\left\langle\sqrt{\rho^{cond}(\vec{r})}\right\rangle_{r_A}\right)^{-1} \tag{S30}$$

$$\rho^{cond}(\vec{r}) = \sum_{\ell,A}\rho_A^{cond}(r_A) \tag{S31}$$

This reshaping is done by minimizing the following optimization functional

$$h^{II}\left(G_A(r_A)\right) = \int_0^{r_{cutoff}}\left[\frac{\left(G_A(r_A) - \rho_A^{wavg}(r_A)\right)^2}{2\sqrt{\rho_A^{wavg}(r_A)}} + \Gamma_A^{II}(r_A)\phi^{II}(r_A)\right]4\pi(r_A)^2\,dr_A - \Phi_A^{II}\varphi_A^{II} \tag{S32}$$

where $\Gamma_A^{II}(r_A)$ and Φ_A^{II} are Lagrange multipliers enforcing constraints (S27) and (S28), respectively.[S3] $h^{II}\left(G_A(r_A)\right)$ is a convex functional with the unique minimum:[S3]

$$G_A(r_A) = \rho_A^{wavg}(r_A) + \sqrt{\rho_A^{wavg}(r_A)}\left(\Phi_A^{II} + \frac{d\Gamma_A^{II}(r_A)}{dr_A} + \frac{2\Gamma_A^{II}(r_A)}{r_A} - \Gamma_A^{II}(r_A)\eta_A^{lower}(r_A)\right) \quad \text{(S33)}$$

During the 4th, 5th, and 6th charge partitioning steps, reshaping is also performed to prevent the tails of buried atoms from becoming too contracted. Specifically, the constraints

$$\phi^{III}(r_A) = \frac{dH_A(r_A)}{dr_A} + \eta_A^{upper}(r_A)H_A(r_A) \geq 0 \quad \text{(S34)}$$

$$\eta_A^{upper}(r_A) = \frac{2.5\ \text{bohr}^{-1}}{\left(1-\left(\tau_A(r_A)\right)^2\right)} \quad \text{(S35)}$$

$$\varphi_A^{III} = \int_0^{r_{cutoff}} \left(H_A(r_A) - G_A(r_A)\right) 4\pi (r_A)^2\, dr_A \Rightarrow 0 \quad \text{(S36)}$$

are imposed by minimizing the following optimization functional (i.e., Lagrangians 7, 10, and 13):

$$h^{III}\left(H_A(r_A)\right) = \int_0^{r_{cutoff}} \left[\frac{\left(H_A(r_A) - G_A(r_A)\right)^2}{2G_A(r_A)} + \Gamma_A^{III}(r_A)\phi^{III}(r_A)\right] 4\pi (r_A)^2\, dr_A - \Phi_A^{III}\varphi_A^{III} \quad \text{(S37)}$$

where $\Gamma_A^{III}(r_A)$ and Φ_A^{III} are Lagrange multipliers enforcing constraints (S34) and (S36), respectively. $h^{III}\left(\sigma_A(r_A)\right)$ is a convex functional whose unique minimum is[S2]

$$H_A(r_A) = G_A(r_A)\left(1 + \Phi_A^{III} + \frac{d\Gamma_A^{III}(r_A)}{dr_A} + \frac{2\Gamma_A^{III}(r_A)}{r_A} - \Gamma_A^{III}(r_A)\eta_A^{upper}(r_A)\right) \quad \text{(S38)}$$

where $H_A^{(5)}$, $H_A^{(6)}$, and $H_A^{(7)}$ are obtained by the reshaping performed on the 4th, 5th, and 6th charge partitioning steps, respectively.

S2. Spin partitioning Lagrangian and flow diagram

S2.1 Spin partitioning Lagrangian

"The electron-spin density can be described using a four component spinor, whose four charge density components are directly related to the total electron density, $\rho(\vec{r})$, and the spin magnetization density, $\vec{m}(\vec{r})$. Either the Dirac spinor or the Pauli spin matrices can be used to describe this spinor.[S4-6] To make our method equally applicable for either spin formulation, we use $\rho(\vec{r})$ and $\vec{m}(\vec{r})$ directly. The operator for measuring the spin of electron j is

$$\vec{s}(j) = \hat{x}s_x(j) + \hat{y}s_y(j) + \hat{z}s_z(j) \quad \text{(S39)}$$

where $s_x(j)$, $s_y(j)$, and $s_z(j)$ are the operators for measuring its spin along the $\hat{x}$, $\hat{y}$, and $\hat{z}$ directions, respectively.[S7] Here, we denote the magnitude and direction of a vector by $b = \left\|\vec{b}\right\|$ and $\hat{b} = \vec{b}/b$, respectively. We use δ^{dirac} to denote the Dirac delta function and δ to denote a variational derivative. For a system containing N electrons, the spin magnetization density, $\vec{m}(\vec{r})$, can be computed by summing the

spins of all electrons at position $\vec{r}$ and dividing by the spin magnitude of an individual electron. Specifically,

$$\vec{m}(\vec{r}) = 2\left\langle \Psi_{el} \left| \sum_{j=1}^{N} \left(\vec{s}(j) \delta^{dirac}\left(\vec{r} - \vec{e}_j\right)\right) \right| \Psi_{el} \right\rangle \quad \text{(S40)}$$

where $\Psi_{el}\left(\left\{\vec{e}_j\right\}\right)$ is the multi-electronic wavefunction and $\left\{\vec{e}_j\right\}$ are the spatial coordinates of the electrons. The factor of 2 occurs because the spin magnitude of an individual electron is one-half. Spin can be measured along any unit direction $\hat{h}$, and the value of the electron-spin density projected onto a measurement direction $\hat{h}$ is

$$\rho\left(\vec{r}, \hat{h}\right) = \left(\rho(\vec{r}) + \vec{m}(\vec{r}) \bullet \hat{h}\right) / 2 \quad \text{(S41)}$$

where $\rho(\vec{r})$ and $\vec{m}(\vec{r})$ are expressed in units of electrons per unit volume. Collinear magnetism occurs when $\hat{m}(\vec{r})$ is parallel to a global magnetization axis, $\hat{h}_{global}$, while non-collinear magnetism occurs when it is not. Collinear magnetism has only two independent electron density components, $\rho^{\alpha}(\vec{r})$ and $\rho^{\beta}(\vec{r})$, which are the electron-spin density projected onto measurement directions $\hat{h}_{global}$ and $-\hat{h}_{global}$, respectively."[S8] For collinear magnetism, the

$$\text{spin\_density}(\vec{r}) = \vec{m}(\vec{r}) \bullet \hat{h}_{global} = \rho^{\alpha}(\vec{r}) - \rho^{\beta}(\vec{r}) \quad \text{(S42)}$$

can take on positive and negative values, depending on whether the density of spin-up or spin-down electrons is greater, respectively. A non-magnetic system has $\vec{m}(\vec{r}) = \vec{0}$ everywhere.

All DDEC methods use the spin partitioning method of Manz and Sholl that applies to both collinear and non-collinear magnetism.[S8] This spin partitioning minimizes the Lagrangian

$$Ж^{spin} = \sum_{A} \left(\oint_{\omega} \oint \rho_A\left(\vec{r}_A, \hat{h}\right) \ln\left(\frac{\rho_A\left(\vec{r}_A, \hat{h}\right)}{w_A^{spin}\left(\vec{r}_A, \hat{h}\right)} \right) d^3\vec{r}_A d^2\omega - \oint \nu_A\left(\vec{r}_A\right) Щ_A\left(\vec{r}_A\right) d^3\vec{r}_A \right) + \int_U \vec{\Lambda}(\vec{r}) \bullet \vec{\Delta}(\vec{r}) d^3\vec{r} \quad \text{(S43)}$$

with respect to the atomic spin magnetization density vectors $\left\{\vec{m}_A\left(\vec{r}_A\right)\right\}$.[S8] Integration over $d^2\omega$ signifies integration over all possible unit vectors $\hat{h}$ that originate at the center of a unit sphere and terminate on the unit sphere's surface.[S8] This Lagrangian is convex and has a unique minimum.[S8] The magnitude is given by

$$m_A\left(\vec{r}_A\right) = \sqrt{\vec{m}_A\left(\vec{r}_A\right) \bullet \vec{m}_A\left(\vec{r}_A\right)} \quad \text{(S44)}$$

The Lagrange multiplier $\nu_A\left(\vec{r}_A\right) \geq 0$ enforces the constraint

$$Щ_A\left(\vec{r}_A\right) = \rho_A\left(\vec{r}_A\right) - m_A\left(\vec{r}_A\right) \geq 0 \quad \text{(S45)}$$

that ensures the assigned $m_A\left(\vec{r}_A\right)$ is chemically feasible.[S8] The Lagrange multiplier $\vec{\Lambda}(\vec{r})$ enforces the constraint

$$\vec{\Delta}(\vec{r}) = \vec{m}(\vec{r}) - \sum_{\ell, A} \vec{m}_A\left(\vec{r}_A\right) \Rightarrow 0 \quad \text{(S46)}$$

that ensures the assigned $\left\{\vec{m}_A\left(\vec{r}_A\right)\right\}$ sum to $\vec{m}(\vec{r})$ at each position $\vec{r}$.[S8] The atomic electron-spin density projected onto a measurement direction $\hat{h}$ is[S8]

$$\rho_A\left(\vec{r}_A,\hat{h}\right)=\left(\rho_A\left(\vec{r}_A\right)+\vec{m}_A\left(\vec{r}_A\right)\bullet\hat{h}\right)\Big/2 \tag{S47}$$

The ASM vectors are defined as

$$\vec{M}_A=\oint\vec{m}_A\left(\vec{r}_A\right)d^3\vec{r}_A \tag{S48}$$

and sum to the total spin magnetic moment of the unit cell

$$\vec{M}=\sum_A\vec{M}_A \tag{S49}$$

The atomic weighting factor for spin partitioning

$$w_A^{spin}\left(\vec{r}_A,\hat{h}\right)=\sqrt{\left(\rho_A^0\left(\vec{r}_A,\hat{h}\right)\right)\left(\rho_A^{avg}\left(r_A,\hat{h}\right)\right)} \tag{S50}$$

optimizes $\rho_A\left(\vec{r}_A,\hat{h}\right)$ to resemble the geometric average of the proportional spin partition $\left(\rho_A^0\left(\vec{r}_A,\hat{h}\right)\right)$ and the spherically averaged spin partition $\left(\rho_A^{avg}\left(r_A,\hat{h}\right)\right)$.[S8] This atomic weighting factor is much more chemically accurate than using purely proportional or purely spherically averaged spin partitioning.[S8] The proportional spin partition assigns spin magnetization density in the same ratio as the electron density:

$$\vec{m}_A^0\left(\vec{r}_A\right)=\vec{m}\left(\vec{r}\right)\rho_A\left(\vec{r}_A\right)\Big/\rho\left(\vec{r}\right) \tag{S51}$$

$$\left(\rho_A^0\left(\vec{r}_A,\hat{h}\right)\right)=\left(\rho_A\left(\vec{r}_A\right)+\vec{m}_A^0\left(\vec{r}_A\right)\bullet\hat{h}\right)\Big/2 \tag{S52}$$

In non-collinear magnetism, the spin magnetization direction $\hat{m}\left(\vec{r}\right)$ varies as a function of position $\vec{r}$. In collinear magnetism, the spin magnetization direction is everywhere parallel to a global spin quantization axis: $\hat{m}\left(\vec{r}\right)\bullet\hat{h}_{global}=\pm1$. DDEC spin partitioning for collinear magnetism is identical to that for non-collinear magnetism, except spin partitioning for collinear magnetism utilizes projections onto the global spin quantization axis, $\hat{h}_{global}$.[S8] For collinear magnetism, the corresponding quantities projected onto the global spin quantization axis are scalars[S8]

$$Ц_A=\vec{M}_A\bullet\hat{h}_{global} \tag{S53}$$

$$Ц=\vec{M}\bullet\hat{h}_{global}=\sum_A Ц_A \tag{S54}$$

Eq. (S53) defines the ASMs, $\{Ц_A\}$, for collinear magnetism. In collinear magnetism, electrons are traditionally referred to as spin-up or spin-down. $Ц_A$ is the number of spin-up minus spin-down electrons assigned to atom A. Ц is the number of spin-up minus spin-down electrons in the unit cell. Because of these scalar projections, collinear spin partitioning requires computing and storing only one-third as many spin components as for non-collinear spin partitioning.[S8] Consequently, required memory and computational time for collinear spin partitioning are about one-third those of non-collinear spin partitioning.

S2.2 Spin partitioning flow diagram

Figure S1 is a flow diagram for DDEC spin partitioning. The equations and sequence of steps follows Manz and Sholl[S8], to which we added parallel computation. Each spin cycle consists of two loops over atoms and grid points. Between these two loops, the ASMs are tallied and checked for convergence. Each of these two loops was parallelized over the outer grid point index.

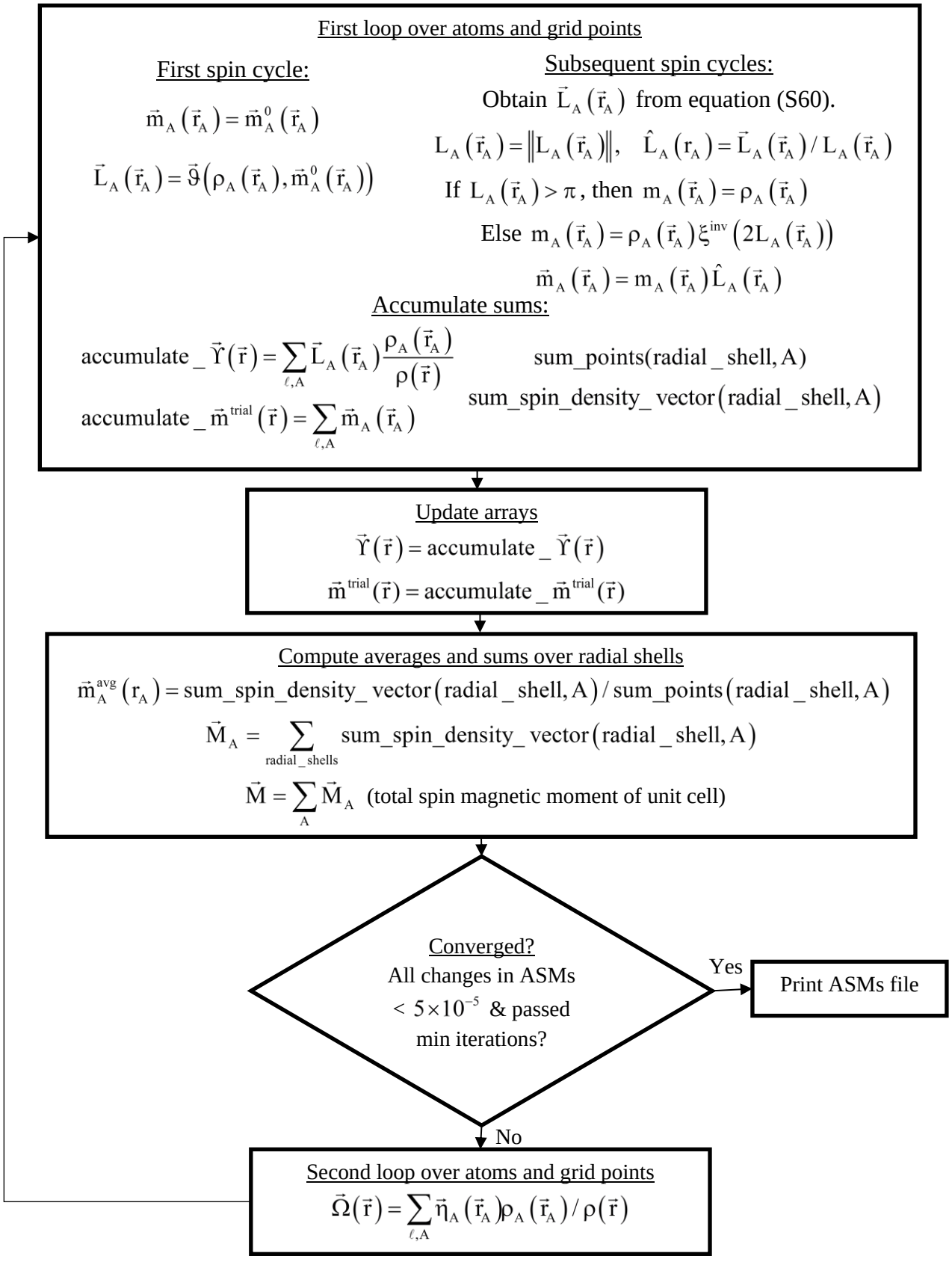

Figure S1. Flow diagram of DDEC spin partitioning.

During each spin cycle, the first loop over grid points computes the $\{\vec{m}_A(\vec{r}_A)\}$ and accumulates $\text{sum\_points}(\text{radial\_shell}, A)$, and $\text{sum\_spin\_density\_vector}(\text{radial\_shell}, A)$ and temporary secondaries of the arrays

$$\vec{\Upsilon}(\vec{r}) = \sum_{\ell,A} \vec{L}_A(\vec{r}_A) \frac{\rho_A(\vec{r}_A)}{\rho(\vec{r})} \tag{S55}$$

$$\vec{m}^{\text{trial}}(\vec{r}) = \sum_{\ell,A} \vec{m}_A(\vec{r}_A) \tag{S56}$$

Each grid point within the cutoff radius is assigned to one of the radial shells of atom A. As the loop over grid points for atom A proceeds, the array sum_points accumulates the sum of all grid point volumes belonging to each radial shell. The array sum_spin_density_vector accumulates $\vec{m}_A(\vec{r}_A)$ times each grid point volume as a sum over all grid points belonging to each radial shell. The sums in Eqs. (S55) and (S56) must be accumulated in temporary secondaries, because the program still needs to access the full values of these arrays (i.e., the primaries) computed from the prior spin cycle to compute $\vec{L}_A(\vec{r}_A)$ in Eq. (S60).

The first spin cycle sets the atomic spin magnetization densities using proportional spin partitioning:

$$\vec{m}_A(\vec{r}_A)\big|_1 = \vec{m}_A^0(\vec{r}_A) \tag{S57}$$

$$\nu_A(\vec{r}_A)\big|_1 = 0 \tag{S58}$$

$$\vec{L}_A(\vec{r}_A)\big|_1 = \vec{\vartheta}\left(\rho_A(\vec{r}_A), \vec{m}_A^0(\vec{r}_A)\right) \tag{S59}$$

During subsequent spin cycles, the atomic spin magnetization densities are computed by solving the following equations

$$\vec{L}_A(\vec{r}_A) = \vec{\Upsilon}(\vec{r}) - \vec{\Omega}(\vec{r}) + \pi \frac{\left(\vec{m}(\vec{r}) - \vec{m}^{\text{trial}}(\vec{r})\right)}{\rho(\vec{r})} + \vec{\eta}_A(\vec{r}_A) \tag{S60}$$

where

$$\vec{\eta}_A(\vec{r}_A) = \left(\vec{\vartheta}\left(\rho_A^{\text{avg}}(r_A), \vec{m}_A^{\text{avg}}(r_A)\right) + \vec{\vartheta}\left(\rho_A(\vec{r}_A), \vec{m}_A^0(\vec{r}_A)\right)\right)/2 \tag{S61}$$

$$\vec{\vartheta}(a, \vec{b}) = \frac{\hat{b}}{2} \xi\left(\frac{b}{a}\right) \tag{S62}$$

$$\xi(\tau) = \pi\left(\left(\tau^{-2} - 1\right)\ln\left(\frac{1-\tau}{1+\tau}\right) + \frac{2}{\tau}\right) \tag{S63}$$

Note that $\xi(0) = 0$ and $\xi(1) = 2\pi$. "The magnitude $L_A(\vec{r}_A)$ was used to update the estimate for $m_A(\vec{r}_A)$ according to two cases. *Case 1:* If $L_A(\vec{r}_A) > \pi$, then $m_A(\vec{r}_A) = \rho_A(\vec{r}_A)$ and $\nu_A(\vec{r}_A) = L_A(\vec{r}_A) - \pi > 0$. *Case 2:* If $L_A(\vec{r}_A) \le \pi$, $m_A(\vec{r}_A) = \rho_A(\vec{r}_A)\xi^{\text{inv}}\left(2L_A(\vec{r}_A)\right)$ and $\nu_A(\vec{r}_A) = 0$. These two cases insure Eq. (S45) is satisfied for all iterations and $\nu_A(\vec{r}_A) \ge 0$."[S8] Then

$$\vec{m}_A(\vec{r}_A) = m_A(\vec{r}_A)\hat{L}_A(\vec{r}_A) \tag{S64}$$

where $\hat{L}_A(\vec{r}_A)$ is the unit direction of $\vec{L}_A(\vec{r}_A)$.

To speed computation, look-up tables were constructed for the $\xi(\tau)$ and ξ^{inv} functions. For convenience, a Fortran module containing these spin-related functions and look-up tables is provided as described in Section S8.

After the first loop over atoms and grid points, $\vec{\Upsilon}(\vec{r})$ and $\vec{m}^{\text{trial}}(\vec{r})$ are updated by copying the values accumulated in the temporary secondaries to the primaries. Then, $\left\{\vec{m}_A^{\text{avg}}(r_A)\right\}$, $\vec{M}_A$, and

$$\vec{M} = \sum_A \vec{M}_A \tag{S65}$$

are computed. After this, a check for ASM convergence is performed. The set of ASMs is converged if the absolute value of each change in ASM component from the previous spin cycle

$$\text{max\_ASM\_change}\big|_j = \max_{\{A\}}\left(\max\left(\left|M_{A,x}\big|_j - M_{A,x}\big|_{j-1}\right|, \left|M_{A,y}\big|_j - M_{A,y}\big|_{j-1}\right|, \left|M_{A,z}\big|_j - M_{A,z}\big|_{j-1}\right|\right)\right) \tag{S66}$$

is less than spin_convergence_tolerance and the minimum number of spin cycles is met. A minimum of seven spin cycles are performed in all cases, unless the system is essentially non-magnetic (i.e., all ASM magnitudes are less than spin_convergence_tolerance) in which case only two spin cycles are performed. If the ASMs are converged, the program breaks and generates the ASMs file; otherwise, the program continues to the next step.

The second major loop over grid points and atoms computes and stores

$$\vec{\Omega}(\vec{r}) = \sum_{\ell,A}\left(\vec{\eta}_A(\vec{r}_A)\frac{\rho_A(\vec{r}_A)}{\rho(\vec{r})}\right) \tag{S67}$$

After this, the program starts the next spin cycle.

For non-collinear magnetism, the CHARGEMOL program also computes and prints the total (Ξ^{tot}), directional (Ξ^{dir}), and magnitude (Ξ^{mag}) changes in $\vec{m}(\vec{r})$ over the unit cell:[S8]

$$\Xi^{\text{tot}} = \Xi^{\text{mag}} + \Xi^{\text{dir}} \geq 0 \tag{S68}$$

$$\Xi^{\text{tot}} = \int_U \sum_{i=1}^{3} \nabla_i \vec{m}(\vec{r}) \bullet \nabla_i \vec{m}(\vec{r}) \, d^3\vec{r} \tag{S69}$$

$$\Xi^{\text{mag}} = \int_U \vec{\nabla} m(\vec{r}) \bullet \vec{\nabla} m(\vec{r}) \, d^3\vec{r} \geq 0 \tag{S70}$$

$$\Xi^{\text{dir}} = \int_U \left(m(\vec{r})\right)^2 \sum_{i=1}^{3} \nabla_i \hat{m}(\vec{r}) \bullet \nabla_i \hat{m}(\vec{r}) \, d^3\vec{r} \geq 0 \tag{S71}$$

These quantities are useful to assess the role of local magnetic torque and spin dynamics in the material. Specifically, the local magnetic torque is negligible when $\Xi^{\text{dir}} \ll \Xi^{\text{mag}}$.[S8] Certain kinds of exchange-correlation functionals only apply in the limit of negligible local magnetic torque.[S8-12]

S3. Equations for bond order analysis

This method of computing bond orders was introduced by Manz.[S13] The first step in bond order analysis is to set the number of exchange components. Spin unpolarized calculations have only one exchange component: the electron density $\rho(\vec{r})$. Collinear magnetism calculations have two exchange components: the electron density and the spin density (Eq. (S42)). Non-collinear magnetism calculations have four exchange components: the electron density and x, y, z components of the spin magnetization

density vector, $\vec{m}(\vec{r})$. The arrays and computational routines in bond order analysis are designed such that exactly the corresponding number of exchange components are allocated in memory and computed.

The second step is to prepare the density grids for bond order analysis by performing the total electron density grid correction described in Section S4 below.

The third step is to compute several quantities involving the local atomic exchange vectors,

$$\vec{\rho}_j^{\,avg}(r_j) = \left(\rho_j^{avg}(r_j), \vec{m}_j^{\,avg}(r_j)\right) \tag{S72}$$

First, the sum of these is computed at each grid point:

$$\vec{\rho}^{\,avg}(\vec{r}) = \sum_{\ell,A} \vec{\rho}_A^{\,avg}(r_A) \tag{S73}$$

Then, the dot product

$$IO(\vec{r}) = \vec{\rho}^{\,avg}(\vec{r}) \bullet \vec{\rho}^{\,avg}(\vec{r}) \tag{S74}$$

is computed and stored for each grid point.

The fourth step is to initialize the bond pair matrix. The bond_pair_matrix is a list of translation symmetry unique atom pairs that have a non-negligible bond order. In periodic materials, the first atom in a translation symmetry unique atom pair can always be taken to be within the reference unit cell. All other atom pairs can be derived through lattice vector translations. Let the first atom in a translation symmetry unique atom pair be denoted by $i = (A,0,0,0)$ indicating atom number A in the reference unit cell. Let the second atom in the pair be denoted by $j = (B,\ell_1,\ell_2,\ell_3)$ indicating a translated image of atom B along the three lattice vectors. Then, the atom pair is symmetry unique if and only if at least one of the following conditions is met: (a) $B > A$, (b) $k_1 > 0$, (c) $k_1 = 0$ and $k_2 > 0$, or (d) $k_1 = k_2 = 0$ and $k_3 > 0$. The translation symmetry unique atom pairs are then further screened to determine whether their density overlap is large enough to merit inclusion in the bond_pair_matrix. The bond_pair_matrix is initialized by first counting the number of atom pairs to include (i.e., num_included_pairs) and then allocating a matrix of size (rows, num_included_pairs). The rows will be used to store relevant information for each included atom pair. First, the following information is stored in the bond_pair_matrix for each included atom pair: (a) $(A,B,\ell_1,\ell_2,\ell_3)$ and (b) the corner points defining a parallelepiped enclosing all grid points that are simultaneously closer than cutoff_radius to atoms i and j. Only grid points simultaneously closer than cutoff_radius to atoms i and j contribute to the contact exchange and overlap population of this atom pair. Sections S7 and S8 of the bond order article described the method for determining included atom pairs and their corresponding parallelepiped corner points for the bond pair matrix.[S13]

The fifth step computes the following quantities by integrating over positions in the relevant parallelepiped for each atom pair in the bond pair matrix: contact exchanges (Eq. (S75)), overlap populations (Eq. (S76)), temp_1 (Eq. (S77)), and temp_2 (Eq. (S78)). For $i \neq j$, the contact exchange (CE) is defined as[S13]

$$CE_{i,j} = 2\oint \frac{\vec{\rho}_i^{\,avg}(r_i) \bullet \vec{\rho}_j^{\,avg}(r_j)}{IO(\vec{r})} \rho(\vec{r})\, d^3\vec{r} \geq 0 \tag{S75}$$

The overlap population is defined as

$$P_{i,j} = \left(2-\delta_{ij}\right)\oint\!\!\oint \frac{\rho_i\left(\vec{r}_i\right)\rho_j\left(\vec{r}_j\right)}{\rho\left(\vec{r}\right)} d^3\vec{r} \geq 0 \tag{S76}$$

The following intermediate terms are computed and stored:

$$temp\_1_{i,j} = \frac{20}{3}\oint\!\!\oint \left(\frac{\vec{\rho}_i^{avg}\left(r_i\right)\bullet\vec{\rho}_j^{avg}\left(r_j\right)}{Ю\left(\vec{r}\right)}\right)^2 \rho\left(\vec{r}\right) d^3\vec{r} \tag{S77}$$

$$temp\_2_{i,j} = 2\oint\!\!\oint \frac{\vec{\rho}_i^{avg}\left(r_i\right)\bullet\vec{\rho}_j^{avg}\left(r_j\right)}{\left(\rho^{avg}\left(\vec{r}\right)\right)^2}\rho\left(\vec{r}\right) d^3\vec{r} \tag{S78}$$

The summed contact exchange (SCE) for atom A

$$SCE_A = \sum_{j\neq i} CE_{i,j} = 2\oint\!\!\oint \frac{\vec{\rho}_A^{avg}\left(r_A\right)\bullet\left(\vec{\rho}^{avg}\left(\vec{r}\right)-\vec{\rho}_A^{avg}\left(r_A\right)\right)}{Ю\left(\vec{r}\right)}\rho\left(\vec{r}\right) d^3\vec{r}_A \tag{S79}$$

is computed by integrating the right-hand side of Eq. (S79) over all grid points having $r_A \leq cutoff\_radius$.[S13]

In Eq. (S76), $\delta_{ij} = 1$ if i and j is the selfsame atom (i.e., the nuclear position of atom i is identical (and not a periodic image) to the nuclear position of atom j) and zero otherwise. The factor of $\left(2-\delta_{ij}\right)$ in Eq. (S76) corresponds to the convention that the number of electrons (N) in the unit cell is given by summing $P_{i,j}$ over all of the atom pairs (including $P_{i,i}$) in the material. Note that (i, j) and (j, i) are considered the same atom pair and included only once in this summation. Some other papers used a slightly different convention, in which the overlap population is defined omitting the $\left(2-\delta_{ij}\right)$ factor.[S14] We believe the convention used here is preferable, because it represents the contribution an atom pair makes to the number of electrons in the unit cell.

We define the average spin polarization of bonding as

$$\zeta_{i,j}^{bond} = \sqrt{\frac{temp\_2_{i,j}}{CE_{i,j}} - 1} \tag{S80}$$

$\zeta_{i,j}^{bond}$ varies between 0 and 1. A non-magnetic system having $\vec{m}\left(\vec{r}\right) = 0$ everywhere will always yield $\zeta_{i,j}^{bond} = 0$. A completely spin-polarized system having $m\left(\vec{r}\right) = \rho\left(\vec{r}\right)$ everywhere will always yield $\zeta_{i,j}^{bond} = 1$. Spin-polarized systems having $0 < m\left(\vec{r}\right) < \rho\left(\vec{r}\right)$ will yield $0 < \zeta_{i,j}^{bond} < 1$.

The sixth step calculates the bond orders using the following equations:[S13]

$$B_{i,j} = CE_{i,j} + \chi_{i,j}^{coord\_num}\chi_{i,j}^{pairwise}\chi_{i,j}^{constraint} \tag{S81}$$

The contact-exchange-weighted coordination number

$$C_A = \left(SCE_A\right)^2 \Big/ \sum_{g\neq i}\left(CE_{i,g}\right)^2 \tag{S82}$$

is used to construct the smooth sigmoidal function

$$\chi_{i,j}^{coord\_num} = 1 - \left(\tanh\left(\left(C_A + C_B - 2\right)/26\right)\right)^2 \tag{S83}$$

The pairwise term is given by

$$\chi_{i,j}^{pairwise} = \min\left(\Omega_{i,j}, CE_{i,j}\right) \tag{S84}$$

$$\Omega_{i,j} = temp\_1_{i,j} + \frac{\left(CE_{i,j}\right)^2}{6} \tag{S85}$$

The constraint term is given by

$$\chi_{i,j}^{constraint} = \min\left(1, \frac{N_A - ½SCE_A}{\sum_{g\neq i}^{coord\_num} \chi_{i,g}^{pairwise}}, \frac{N_B - ½SCE_B}{\sum_{g\neq j}^{coord\_num} \chi_{j,g}^{pairwise}}\right) \tag{S86}$$

In this step, each atom's sum of bond orders (SBO) is computed from the right-hand side of the following equation:

$$SBO_A = \sum_{j\neq i} B_{i,j} = \sum_{(i,j)\in BPM} B_{i,j} + \left[SCE_A - \sum_{(i,j)\in BPM} CE_{i,j}\right] \tag{S87}$$

where $(i,j) \in BPM$ denotes corresponding atom pairs in the bond pair matrix. In Eq. (S87), the bracketed term approximately includes all atom pairs, even those not explicitly included in the bond_pair_matrix.[S13]

Finally, the computed bond orders, SBOs, and overlap populations are printed. For convenience, only those bond orders greater than BO_print_threshold (e.g., 0.001) are printed. The atom pairs and bonds are listed in order to make them easy to locate. For each atom pair, the first atom is always located in the reference unit cell and the second atom includes translation indices. For example, (-1, 0, 2) indicates that the second atom is translated by -1 along the first lattice vector, is not translated along the second lattice vector, and is translated by +2 along the third lattice vector. For non-periodic materials, all of the translation indices will obviously be (0, 0, 0), because there are no periodic translations. It is important to keep in mind that an atom can be bonded to a translated image of itself. For example, in the Ni fcc crystal, which contains only one atom in the unit cell, each Ni atom has DDEC6 bond order of 0.28 with each of its nearest neighbors (which are translated images of itself). For each printed bond order, the spin polarization of bonding (Eq. (S80)) is also printed. Atom pair overlap populations are printed to another file. Overlap populations are printed for every atom pair in the bond pair matrix.

S4. Algorithm for total electron density grid correction

S4.1 Overview

Total electron density grid correction is a process that corrects the $\{\rho(\vec{r})\}$ grid to intrinsically include the valence occupancy corrections. During charge partitioning, the valence occupancy corrections are added extrinsically (i.e., external to the $\{\rho(\vec{r})\}$). After total electron density grid correction, the valence occupancy corrections do not need to be added extrinsically, because the direct integration of $\{\rho(\vec{r})\}$ using the grid already includes them. In this context "valence occupancy correction" refers to the occupancy correction for the number of electrons, N_A, assigned to each atom. The occupancy corrections for atomic dipoles, quadrupoles, and spin moments are not included in the total electron density grid correction. Total electron density grid correction does not change the number of electrons, N_A, or the net charge, q_A, assigned to each atom.

The primary reason for including total electron density grid correction is that it allows $\{\rho_A(\vec{r}_A)\}$ to be integrated to yield N_A without needing to externally add an occupancy correction. This is critical for evaluating quantities that are nonlinear functionals of $\{\rho_A(\vec{r}_A)\}$. Bond orders quantify the number of electrons exchanged between two atoms. This requires that $\rho_A(\vec{r}_A)$ integrate to the correct number of electrons—hence the need for a total electron density grid correction.

The scheme described here for total electron density grid correction is similar to that used for core grid correction, with a few important differences. The core grid correction was described in the Electronic Supplementary Information of our previous article.[S2] The total electron density grid correction does not replace core grid correction, but acts in addition to it. That is, core grid correction is performed during the core electron partitioning and total electron density grid correction is performed at the start of bond order analysis. Two key differences between core grid correction and total electron density grid correction are: (a) the atomic weighting factors remain fixed during the total electron density grid correction but are updated during core grid correction and (b) core grid correction is applied to all grid points while total electron density grid correction is applied only to 125 grid points surrounding each nucleus (i.e., a 5×5×5 block).

If using variable-spaced atom-centered integration grids[S15-16] instead of the uniform integration grids used here, then this total electron density grid correction would be unnecessary.

S4.2 Design criteria

a) The NAC assigned to each atom is not affected by the total electron density grid correction. Specifically, the NAC before and after total electron density grid correction will be the same to within a specified convergence tolerance (e.g., 10^{-5} e).
b) The total electron density grid correction should never produce a negative electron density for any grid point.
c) For a particular atom, the total electron density grid correction should not change the relative ordering of grid point densities. Specifically, if grid point 1 contains a higher electron density assigned to atom A than grid point 2, then after the correction is applied this should still be the case.
d) Because nuclear cusps contribute most of the integration error, the total electron density grid correction should be localized to those grid points close to atomic nuclei.

S4.3 Iterative algorithm

As shown in Figure S2, the following sequence of steps is performed to correct the total electron density grid:

a) In each correction iteration i, the target correction for each atom is computed as following:

$$\text{Correction}_A\big|_i = N_A - \oint \rho_A^{avg}(r_A)\big|_i \, d^3\vec{r}_A \qquad \text{(S88)}$$

Eq. (S88) may seem strange at first. The key to understanding it is to note that N_A remains fixed throughout the entire total electron density grid correction procedure, while $\rho_A^{avg}(r_A)\big|_i$ is updated during each correction iteration, i.

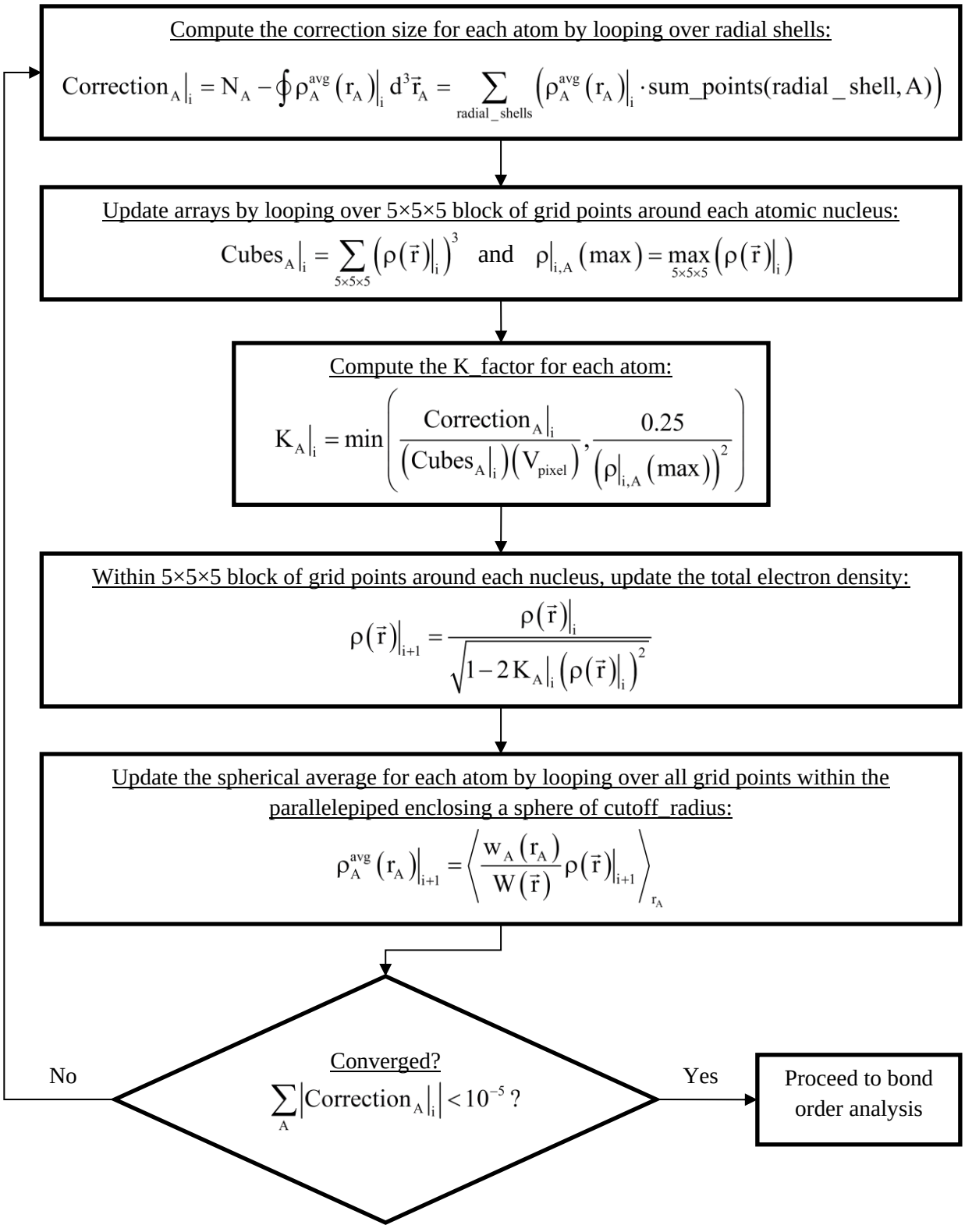

Figure S2: Flow diagram of total electron density grid correction.

b) The following quantities are computed for the 5×5×5 block of grid points centered around each atom A:

$$\mathrm{Cubes_A}\big|_i = \sum_{5\times5\times5} \left(\rho(\vec{r})\big|_i\right)^3 \tag{S89}$$

$$\rho\big|_{i,A}(\max) = \max_{5\times5\times5}\left(\rho(\vec{r})\big|_i\right) \tag{S90}$$

c) A real variable K_A is computed for each atom using the following equations

$$K_A\big|_i = \min\left(\frac{\mathrm{Correction_A}\big|_i}{\left(\mathrm{Cubes_A}\big|_i\right)\left(V_{pixel}\right)}, \frac{0.25}{\left(\rho\big|_{i,A}(\max)\right)^2}\right) \tag{S91}$$

where V_{pixel} is the volume of each grid point.

d) Within the 5×5×5 block of grid points surrounding each atom, the total electron density grid, $\rho(\vec{r})$, is updated by

$$\rho(\vec{r})\big|_{i+1} = \frac{\rho(\vec{r})\big|_i}{\sqrt{1-2K_A\big|_i\left(\rho(\vec{r})\big|_i\right)^2}} \tag{S92}$$

e) $\rho_A^{avg}(r_A)$ is updated by

$$\rho_A^{avg}(r_A)\big|_{i+1} = \left\langle \frac{w_A(r_A)}{W(\vec{r})}\rho(\vec{r})\big|_{i+1}\right\rangle_{r_A} \tag{S93}$$

If $\sum_A \left|\mathrm{Correction_A}\big|_i\right| < \mathrm{charge\_convergence\_tolerance}$, the calculation is considered converged and exits. Otherwise, the calculation goes back to step a) and repeats the sequence for the next iteration (i.e., iteration i+1).

S4.4 Proof this iterative algorithm satisfies the design criteria

a) <u>The iterative scheme converges to the desired solution</u>. *Proof*: Near the solution, we have

$$K_A\big|_i = \left(\frac{\mathrm{Correction_A}\big|_i}{\left(\mathrm{Cubes_A}\big|_i\right)\left(V_{pixel}\right)}\right) \text{ and } \left|K_A\big|_i\left(\rho(\vec{r})\big|_i\right)^2\right| \ll 1 \tag{S94}$$

Therefore, we can expand Eq. (S92) as a Taylor series to give

$$\rho(\vec{r})\big|_{i+1} = \left(\rho(\vec{r})\big|_i + \left(\frac{\mathrm{Correction_A}\big|_i}{\left(\mathrm{Cubes_A}\big|_i\right)\left(V_{pixel}\right)}\right)\left(\rho(\vec{r})\big|_i\right)^3\right) + \mathrm{residual\_1} \tag{S95}$$

Summing Eq. (S95) over the 5×5×5 block of grid points yields

$$\sum_{5\times5\times5}\left(\rho(\vec{r})\big|_{i+1}\right) = \sum_{5\times5\times5}\left(\rho(\vec{r})\big|_i\right) + \frac{\mathrm{Correction_A}\big|_i}{V_{pixel}} + \mathrm{residual\_2} \tag{S96}$$

Multiplying both sides of Eq. (S96) times V_{pixel} and substituting Eq. (S88) gives

$$\oint \rho_A^{avg}(r_A)\big|_{i+1}\, d^3\vec{r}_A = N_A + \mathrm{residual\_3} \tag{S97}$$

Since the residuals contain higher order terms that tend rapidly towards zero as $\text{Correction}_A\big|_i$ tends towards zero, convergence will be achieved when

$$\text{Correction}_A\big|_i \to 0 \tag{S98}$$

which is the desired solution.

b) The corrected total electron density is nonnegative at every grid point. *Proof*: The minimum of the factor $\sqrt{1-2K_A\big|_i\left(\rho_A\left(\vec{r}_A\right)\big|_i\right)^2}$ occurs when $K_A\big|_i > 0$ and for the grid point $\rho_A\left(\vec{r}_A\right)\big|_i(\max)$. From Eq. (S91), it follows $K_A\big|_i\left(\rho_A\left(\vec{r}_A\right)\big|_i\right)^2 \le 0.25$. Therefore, $\sqrt{1-2K_A\big|_i\left(\rho_A\left(\vec{r}_A\right)\big|_i\right)^2} \ge \sqrt{1/2}$. Examining Eq. (S92), this means $\rho\left(\vec{r}\right)\big|_{i+1} \ge 0$.

c) For a particular atom, the correction does not change the relative ordering of grid point densities. *Proof*: Consider the function

$$ъ(s) = \frac{s}{\sqrt{1-2Ks^2}} \tag{S99}$$

which has the derivative

$$\frac{dъ}{ds} = \frac{1}{\left(1-2Ks^2\right)^{3/2}} \tag{S100}$$

$ъ(s)$ is a monotonically increasing function of s over the range $Ks^2 < 0.5$. Because Eq. (S92) has the functional form $ъ(s)$, the relative ordering of grid point densities is preserved for each atom.

d) The correction is localized to those grid points closest to atomic nuclei. This is manifestly true, because the correction is limited to a 5×5×5 block of grid points around each atomic nucleus.

S4.5 What features cause this scheme to converge rapidly and robustly?

a) The correction is localized to regions near each atomic nucleus where typically $w_A\left(r_A\right)/W\left(\vec{r}\right) \approx 1$. This makes corrections for different atoms almost independent of each other.

b) The relative ordering of electron density values for an atom is preserved. This ensures a smooth behavior.

c) Convergence is rapid near the solution, as evidenced by the Taylor series expansion in Eqs. (S95)–(S97).

d) Examining Eqs. (S91) and (S92), the density changes are bounded by

$$\frac{1}{\sqrt{3}} \le \frac{\rho\left(\vec{r}\right)\big|_{i+1}}{\rho\left(\vec{r}\right)\big|_i} \le \sqrt{2} \tag{S101}$$

The extreme values occur for the grid point corresponding to $\rho\big|_{i,A}(\max)$ when $\sum_{5\times5\times5}\left(\rho\left(\vec{r}\right)\big|_i\right)^3$ is dominated by $\rho\big|_{i,A}(\max)$ such that $\sum_{5\times5\times5}\left(\rho\left(\vec{r}\right)\big|_i\right)^3 \approx \left(\rho\big|_{i,A}(\max)\right)^3$ and under the condition that $\left|\text{Correction}_A\big|_i\right|$ is large. Under these conditions, $K_A > 0$ gives the limiting behavior

$$\rho(\vec{r})\big|_{i+1,A}(\max) \approx \frac{\rho(\vec{r})\big|_{i,A}(\max)}{\sqrt{1-2\dfrac{0.25}{\left(\rho(\vec{r})\big|_{i,A}(\max)\right)^2}\left(\rho(\vec{r})\big|_{i,A}(\max)\right)^2}} \approx \sqrt{2}\,\rho(\vec{r})\big|_{i,A}(\max) \qquad \text{(S102)}$$

Under these conditions, $K_A < 0$ gives the limiting behavior

$$\rho(\vec{r})\big|_{i+1,A}(\max) \approx \frac{\rho(\vec{r})\big|_{i,A}(\max)}{\sqrt{1-2\dfrac{\text{Correction}_A\big|_i}{\left(\rho(\vec{r})\big|_{i,A}(\max)\right)^3 V_{pixel}}\left(\rho(\vec{r})\big|_{i,A}(\max)\right)^2}} \qquad \text{(S103)}$$

With $K_A < 0$ and $\text{Correction}_A\big|_i$ dominated by $\rho(\vec{r})\big|_{i,A}(\max)$, substituting $\text{Correction}_A\big|_i \approx -\left(\rho(\vec{r})\big|_{i,A}(\max)\right)\left(V_{pixel}\right)$ into Eq. (S103) gives:

$$\rho(\vec{r})\big|_{i+1,A}(\max) \approx \frac{\rho_A(\vec{r}_A)\big|_i(\max)}{\sqrt{3}} \qquad \text{(S104)}$$

Noting that $\left(\sqrt{2}\right)^{20} = 1024$, this means about 20 iterations are required to increase a grid point density by a factor of 10^3. Noting that $\left(\sqrt{3}\right)^{13} = 1262.665$, this means about 13 iterations are required to decrease a grid point density by a factor of 10^3. Because the approach to convergence is smooth and the total electron density assigned to each grid point is never off by more than a factor of 10^6, this means convergence is always achieved in fewer than 40 iterations. In practice, convergence is nearly always achieved in fewer than 20 iterations.

S5. Allocation and deallocation of big arrays

Figure S3 shows the allocation and deallocation of big arrays. Big arrays are those that run over the grid points in the unit cell. The name of the big array is listed in the first column. The remaining columns list the modules where big arrays exist. The modules are listed in sequence from left to right, with the colored strip for each array indicating those modules for which the array exists. The array is allocated and initialized in the left-most module of the colored strip and deallocated in the right-most module of the colored strip.

Names of the big arrays are explained as follows. The entries labeled “raw grid inputs” refer to arrays that store data read from a file before it has been parsed and transferred into more formal arrays. The word “pseudodensity” just means “a density-like quantity”. The valence and spin interpolation grids refer to temporary grids of different spacing than the main grids. Arrays beginning with the term “accumulate” are temporary secondaries that compute the updated value of an array through summation. These are used when access to the previously-computed primary array is needed during the update computation. After the value of the temporary secondary has been completely updated, its updated value is transferred to the primary array. Names with the word “projection” in them refer to the scalar projection of a vector quantity onto $\hat{h}_{global}$ in collinear magnetism. The maximum_dominant_atom_weight stores the $\max_{\{A\}}\left(w_A(r_A)\right)$ for each grid point, and dominant_atom_points stores the atom number yielding this

maximum value for each grid point. Table S1 shows the correspondence between selected names of big arrays in CHARGEMOL and their equivalent mathematical notation in this paper.

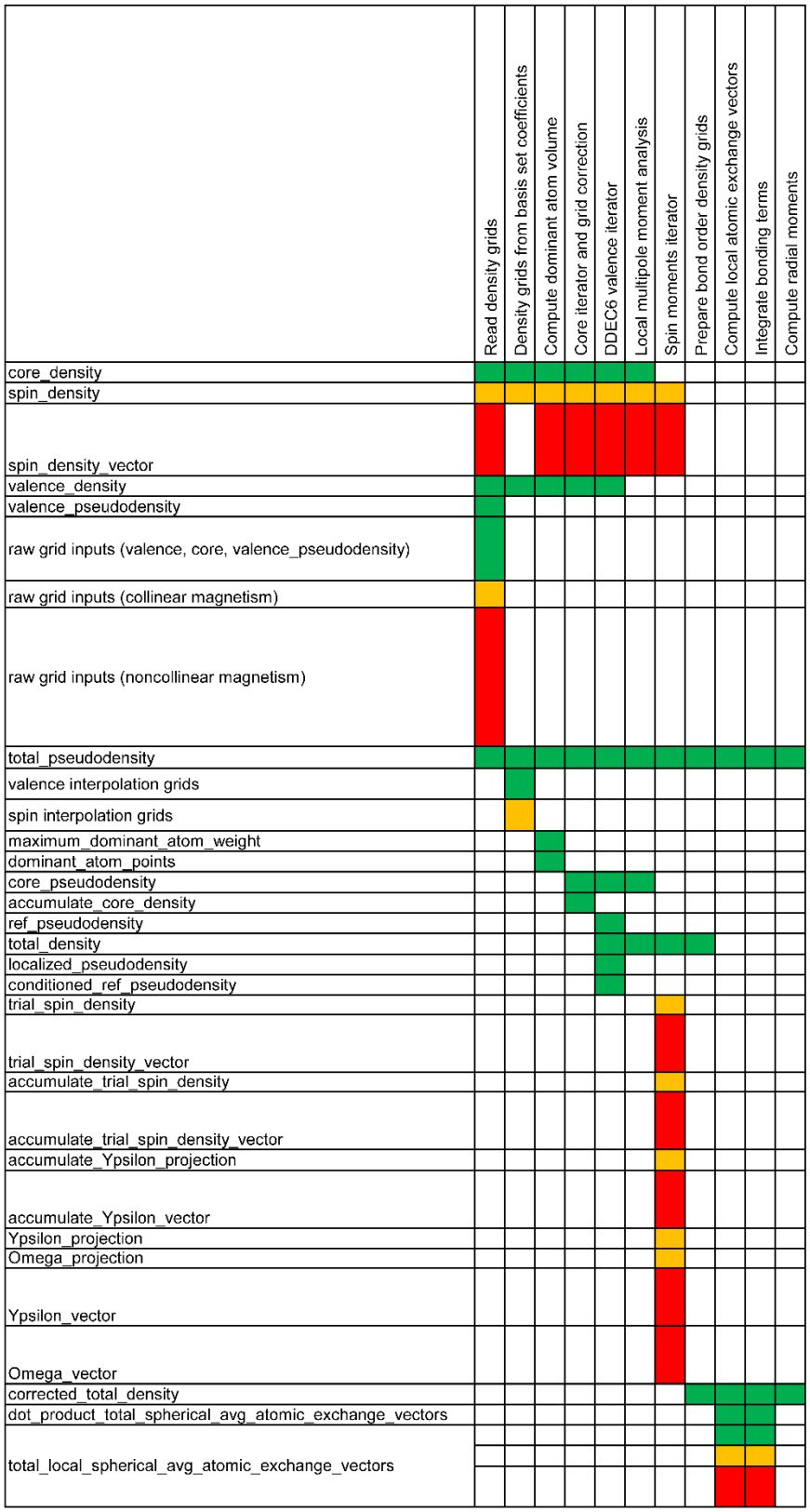

Figure S3. Modules with allocation and deallocation of big arrays.

Table S1. Names of selected big arrays in CHARGEMOL and their equivalent mathematical notation.

array name	mathematical notation
core_density	$\rho^{\text{core}}(\vec{r})$
spin_density	$\vec{m}(\vec{r})\bullet\hat{h}_{\text{global}} = \rho^{\alpha}(\vec{r}) - \rho^{\beta}(\vec{r})$
spin_density_vector	$\vec{m}(\vec{r})$
valence_density	$\rho^{\text{val}}(\vec{r})$
valence_pseudodensity	[a]
total_pseudodensity	$W(\vec{r})$
core_pseudodensity	$W^{\text{core}}(\vec{r})$
ref_pseudodensity	$\rho^{\text{ref}}(\vec{r})$
total_density	$\rho(\vec{r})$
localized_pseudodensity	$\sum_{\ell,A}\left(\bar{\rho}_A^{\text{ref}}\left(r_A, q_A^{\text{ref}}\right)\right)^4$
conditioned_ref_pseudodensity	$\rho^{\text{cond}}(\vec{r})$
trial_spin_density	$\sum_{\ell,A}\left(\rho_A^{\alpha}(\vec{r}) - \rho_A^{\beta}(\vec{r})\right)$
trial_spin_density_vector	$\vec{m}^{\text{trial}}(\vec{r})$
Ypsilon_projection	$\vec{\Upsilon}(\vec{r})\bullet\hat{h}_{\text{global}}$
Omega_projection	$\vec{\Omega}(\vec{r})\bullet\hat{h}_{\text{global}}$
Ypsilon_vector	$\vec{\Upsilon}(\vec{r})$
Omega_vector	$\vec{\Omega}(\vec{r})$
corrected_total_density	$\rho(\vec{r})$
dot_product_total_spherical_avg_atomic_exchange_vectors	$\text{Ю}(\vec{r})$
total_local_spherical_avg_atomic_exchange_vectors	$\vec{\rho}^{\text{avg}}(\vec{r})$

[a] Smooth valence pseudodensity in PAW method.

In Figure S3, the height of each cell is proportional to the memory required to store that array, with a unit height corresponding to storing one double precision real number at every grid point. Green colored cells are used by non-magnetic, collinear magnetic, and non-collinear magnetic calculations. Yellow colored cells are used only by collinear magnetic calculations. Red colored cells are used only by non-collinear magnetic calculations. The total_local_spherical_avg_atomic_exchange_vectors has a height proportional to the number of exchange components: 1 for non-magnetic, 2 for collinear magnetic, and 4 for non-collinear magnetic calculations.

Some calculations will not use all modules. Calculations with pre-computed electron density grid input files will use the Read density grids module, while calculations with Gaussian basis set coefficients input files will used the Density grids from basis set coefficients module. Non-magnetic calculations skip the spin moments iterator. The Compute dominant atom volume module is used for calculations that compute the valence occupancy corrections by processing both the valence density and valence pseudodensity input files of a projector augmented wave (PAW) quantum chemistry calculation.

One should be careful about interpreting the total memory requirements from Figure S3. Specifically, some big arrays might be deallocated in a module before other big arrays are allocated in that same module. This can cause the overall memory requirements for that module to be smaller than the total number of big arrays used in that module. Localized_pseudodensity does not increase the memory requirements of the DDEC6 valence iterator, because it is used and deallocated before some of the other big arrays are allocated.

One should keep in mind that Figure S3 represents a snapshot of big array usage in the CHARGEMOL program at the time this article was first published. Because software programs evolve, precise treatment of big arrays in this software program may change over time. Also, only those modules in which big arrays are allocated or deallocated are listed in Figure S3. The program contains many additional modules which are not listed, because they neither allocate nor deallocate big arrays.

S6. Computational parameters

Table S2 shows the parameters used by CHARGEMOL to ensure computational efficiency and precision. Double precision was kept during the entire program, making the first 15 digits after the decimal point significant. A cutoff radius was set to 500 picometers for all atoms, because the atomic electron density after that radius is negligible. Integrations over radius were obtained by dividing the 500 picometers into 100 uniformly spaced radial shells and adding the contribution of each shell. To avoid division by zero errors, we used if statements that avoid divisions for denominators less than a zero_tolerance of 10^{-10}. To help ensure accurate integrations, each grid point contributes less than 0.03 valence electrons and a volume of less than 0.0157 bohr^3. An integration tolerance and an integration tolerance percent (i.e., the larger of 0.1 electrons and 0.10%) was set as a maximum allowed error on the integrated number of (valence) electrons. The calculation will terminate with a message to use a better grid if these criteria are not met.

Table S2 also shows the parameters used to compute the NACs. The charge_convergence _tolerance was set to 10^{-5}, meaning the NACs have to change less than that in two consecutive iterations to be considered converged during the update_kappa = .TRUE. iterations. During update_kappa = .TRUE. iterations, κ_A is set to 0 for all atoms that do not overlap other atoms, and an atom is considered to not overlap any other atoms if $\partial N_A/\partial\kappa_A \le \text{nonoverlapping\_atom\_tolerance}$.[S2] After charge partitioning completes, linear regression over $\text{rmin\_cloud\_penetration} \le r_A \le \text{cutoff\_radius}$ is used to compute the electron cloud parameters д and Ъ for each atom.

The ASMs were computed using the following parameters. A spin_convergence_tolerance of 5×10^{-5} was used. Xi_threshold and Xi_zero_tolerance define thresholds below which the Xi, Xi_derivative, and Xi_inverse functions were computed using linear interpolation to avoid division by zero errors. Lookup tables with num_lookup_points = 10000 points were constructed to more quickly evaluate the spin-related mathematical functions and their inverses.

Finally, Table S2 shows the bond order analysis parameters. Bond orders smaller than 0.001 were not printed.

Table S2. Parameters for the DDEC NACs, ASMs, and bond orders calculations.

variable	value	units	function
		global parameters	
dp	8	bytes	First 15 digits are significant
nshells	100	shells	Number of radial integration shells
zero_tolerance	10^{-10}	none	Helps to avoid divisions by zero
integration_tolerance & integration_tolerance_percent	0.1 0.10	electrons percent	Number of (valence) electrons must integrate with a difference less than the larger of these
pixel_integration_tolerance	0.03	electrons	Each pixel should contribute fewer valence electrons than this
maxpixelvolume	0.0157	$bohr^3$	Volume per grid point cannot exceed this
cutoff_radius	500	picometers	Atomic electron density assumed to be zero outside this radius
		net atomic charges	
charge_convergence_tolerance	10^{-5}	electrons	Each NAC must change less than this to converge
nonoverlapping_atom_tolerance	10^{-7}	electrons	Defines the $\partial N_A/\partial \kappa_A$ threshold for a non-overlapping atom
rmin_cloud_penetration	200	picometers	The electron cloud parameters are fit from this radial value to cutoff_radius
		atomic spin moments	
spin_convergence_tolerance	$5x10^{-5}$	electrons	Each ASM must change less than this to converge
Xi_threshold	0.001	none	For smaller τ, use linear interpolation to compute Xi and Xi derivative functions
Xi_zero_tolerance	10^{-6}	none	For smaller ξ, use linear interpolation to compute ξ^{inv} function
num_lookup_points	10000	none	Number of values in spin look up tables
		bond orders	
BO_print_cutoff	0.001	none	Smaller bond orders are not printed

S7. How to use the enclosed reshaping subroutines

Within the module_reshaping_subroutines.f08 file are two Fortran subroutines for reshaping functions of r_A. All of the real numbers used in these routines are double precision (i.e., 64 bit, 8 byte).

The SUBROUTINE monotonic_decay_subroutine performs reshaping to create a monotonically decreasing function of r_A:

```
PURE SUBROUTINE monotonic_decay_subroutine(unreshaped_partial_density, nshells,
radial_shell_integration_weight, reshaped_partial_density, local_reshaping_iterations)
INTEGER, INTENT(IN) :: nshells
REAL(kind=dpr), INTENT(IN), DIMENSION(:) :: unreshaped_partial_density,
radial_shell_integration_weight
REAL(kind=dpr), INTENT(OUT), DIMENSION(:) :: reshaped_partial_density
INTEGER, INTENT(OUT) :: local_reshaping_iterations
```

The unreshaped_partial_density array contains the input function on each radial shell. The number of radial shells is called nshells. The reshaped_partial_density is the output function that has been constrained to monotonically decrease by minimizing the reshaping Lagrangian h^{I} described in Eq. (S22) above. (The output will always be non-negative: reshaped_partial_density ≥ 0.) This Lagrangian constrains the integrals of the input (i.e., unreshaped_partial_density) and output (i.e., reshaped_partial_density) functions over the grid points to be equal. These integrals are computed using

$$\oint f\left(r_A\right) d^3\vec{r}_A = \int_0^{r_{cutoff}} 4\pi\left(r_A\right)^2 f\left(r_A\right) dr_A = \sum_{i=1}^{nshells} \text{Ц}_i f_i \qquad \text{(S105)}$$

where f_i is the value of the function $f(r_A)$ on the i^{th} radial shell and Ц_i is the integration weight for that radial shell. These radial shell integration weights $\{\text{Ц}_i\}$ are stored in the array radial_shell_integration_weight that is an input argument of the subroutine. Note that the radial_shell_integration_weight and function value are expected to be positive. If $\text{Ц}_i \leq 0$ or $f_i \leq 0$, that radial shell will be ignored when enforcing the monotonicity constraint (but will still be included in the integration of the function, Eq. (S105)). The output variable local_reshaping_interations is the integer number of reshaping iterations that were required to achieve convergence.

The SUBROUTINE tail_exponential_decay_subroutine performs reshaping to create a function of r_A that satisfies the constraints:

$$f_{i-1} * single\_first\_exp\_const_i \geq f_i \geq f_{i-1} * single\_second\_exp\_const_i \qquad \text{(S106)}$$

The two constraints in Eq. (S106) are applied separately and recursively, starting with i = 2 and continuing until i = nshells. The inputs should be constructed by

$$single\_first\_exp\_const_i = \exp\left(-\eta_A^{lower}\left(r_A\right) * \Delta r_A\right) \qquad \text{(S107)}$$

$$single\_second\_exp\_const_i = \exp\left(-\eta_A^{upper}\left(r_A\right) * \Delta r_A\right) \qquad \text{(S108)}$$

where Δr_A is the distance between radial shell i and radial shell (i-1). (Note that Δr_A can be different for different values of i.) This subroutine has the form:

```
PURE SUBROUTINE tail_exponential_decay_subroutine(unreshaped_partial_density,
single_first_exp_const, single_second_exp_const, nshells, radial_shell_integration_weight,
reshaped_partial_density, local_reshaping_iterations)
INTEGER, INTENT(IN) :: nshells
REAL(kind=dpr), INTENT(IN), DIMENSION(:) :: single_first_exp_const, single_second_exp_const
REAL(kind=dpr), INTENT(IN), DIMENSION(:) :: unreshaped_partial_density,
radial_shell_integration_weight
REAL(kind=dpr), INTENT(OUT), DIMENSION(:) :: reshaped_partial_density
INTEGER, INTENT(OUT) :: local_reshaping_iterations
```

The unreshaped_partial_density array contains the input function on each radial shell. The number of radial shells is called nshells. The reshaped_partial_density is the output function that has been constrained to satisfy Eq. (S106) by minimizing the reshaping Lagrangians h^{II} and h^{III} described in Eqs. (S32) and (S37). (The output will always be non-negative: reshaped_partial_density ≥ 0.) These Lagrangians

constrain the integrals of the input (i.e., unreshaped_partial_density) and output (i.e., reshaped_partial_density) functions over the grid points to be equal. These integrals are computed using Eq. (S105) described above. The radial shell integration weights $\{\text{Ш}_i\}$ are stored in the array radial_shell_integration_weight that is an input argument of the subroutine. Note that the radial_shell_integration_weight and function value are expected to be positive. If $\text{Ш}_i \le 0$ or $f_i \le 0$, that radial shell will be ignored when enforcing the constraint preventing the function from being too diffuse (but will still be included in the integration of the function, Eq. (S105)). The output variable local_reshaping_interations is the integer number of reshaping iterations that were required to achieve convergence.

S8. How to use the enclosed spin functions

Within the module_spin_functions.f08 file are several spin-related functions and subroutines written in Fortran. All of the real numbers used in these routines are double precision (i.e., 64 bit, 8 byte).

The following three functions are elemental functions, which means they can work on each element of an array argument as if it were a scalar. All three of these functions have double precision real number arguments and return a double precision real number result:

ELEMENTAL FUNCTION calculate_Xi(tau) RESULT(Xi) → $\xi(\tau)$

ELEMENTAL FUNCTION calculate_Xi_derivative(tau) RESULT(Xi_derivative) → $d\xi / d\tau$

ELEMENTAL FUNCTION calculate_inverse_Xi(Xi_value) RESULT(inv_Xi) → $\xi^{inv}(\text{Xi\_value})$

To speed computation, the SUBROUTINE generate_spin_lookup_tables() can be called to generate lookup tables for calculate_Xi and calculate_inverse_Xi. These lookup tables are called Xi_lookup and inverse_Xi_lookup, respectively. A lookup table for calculate_Xi_derivative was not computed, because this function was only used to calculate_inverse_Xi which is used to generate its own lookup table. We used 10000 lookup points.

The value of each spin function can be evaluated quickly via the following functions performing interpolation using these lookup tables. Each of these two functions return a double precision real number as the function result, and the result variable equals the function name. For clarity, the declaration statements for the input arguments are listed below:

```
PURE FUNCTION fast_calculate_Xi(tau, Xi_lookup)
REAL(kind=dpr), INTENT(IN) :: tau, Xi_lookup(num_lookuppoints)

PURE FUNCTION fast_calculate_inverse_Xi(Xi_value, inverse_Xi_lookup)
REAL(kind=dpr), INTENT(IN) :: Xi_value, inverse_Xi_lookup(num_lookuppoints)
```

Finally, the above functions are used to compute $\vec{\vartheta}(a,\vec{b}) = \frac{\hat{b}}{2}\xi\left(\frac{b}{a}\right)$ for non-collinear magnetism (where $b = \|\vec{b}\|$ is the magnitude of $\vec{b}$), whose result variable equals the function name:

```
PURE FUNCTION calculate_theta_vector(a, b_vector, Xi_lookup)
REAL(kind=dpr), INTENT(IN) :: a, b_vector(3), Xi_lookup(num_lookuppoints)
REAL(kind=dpr) :: calculate_theta_vector(3)
```

For collinear magnetism, the corresponding function is called calculate_theta_scalar, and it returns the scalar projection of $\vec{\vartheta}\left(a,\vec{b}\right)$ onto $\hat{h}_{global}$ into a result variable equal to the function name:

```
PURE FUNCTION calculate_theta_scalar(a, b_projection, Xi_lookup)
REAL(kind=dpr), INTENT(IN) :: a, b_projection, Xi_lookup(num_lookuppoints)
REAL(kind=dpr) :: calculate_theta_scalar
```

Here, the input argument b_projection is the scalar projection of $\vec{b}$ onto $\hat{h}_{global}$ (i.e., $b\_projection = \vec{b} \bullet \hat{h}_{global}$).

References:

S1. R. F. Nalewajski and R. G. Parr, *Proc. Natl. Acad. Sci. U.S.A.*, 2000, **97**, 8879-8882.
S2. T. A. Manz and N. Gabaldon Limas, *RSC Adv.*, 2016, **6**, 47771-47801.
S3. T. A. Manz and D. S. Sholl, *J. Chem. Theory Comput.*, 2012, **8**, 2844-2867.
S4. P. A. M. Dirac, *Proc. Roy. Soc. London Ser. A*, 1928, **117**, 610-624.
S5. P. A. M. Dirac, *Proc. Roy. Soc. London Ser. A*, 1928, **118**, 351-361.
S6. L. L. Foldy and S. A. Wouthuysen, *Phys. Rev.*, 1950, **78**, 29-36.
S7. A. Szabo and N. Ostlund, *Modern Quantum Chemistry*, Dover, Mineola, NY, 1996, pp 97-107.
S8. T. A. Manz and D. S. Sholl, *J. Chem. Theory Comput.*, 2011, **7**, 4146-4164.
S9. I. W. Bulik, G. Scalmani, M. J. Frisch and G. E. Scuseria, *Phys. Rev. B*, 2013, **87**, 035117.
S10. G. Scalmani and M. J. Frisch, *J. Chem. Theory Comput.*, 2012, **8**, 2193-2196.
S11. K. Capelle, G. Vignale and B. L. Gyorffy, *Phys. Rev. Lett.*, 2001, **87**, 206403.
S12. S. Sharma, J. K. Dewhurst, C. Ambrosch-Draxl, S. Kurth, N. Helbig, S. Pittalis, S. Shallcross, L. Nordstrom and E. K. U. Gross, *Phys. Rev. Lett.*, 2007, **98**, 196405.
S13. T. A. Manz, *RSC Adv.*, 2017, **7**, 45552-45581.
S14. I. Mayer and P. Salvador, *Chem. Phys. Lett.*, 2004, **383**, 368-375.
S15. A. D. Becke, *J. Chem. Phys.*, 1988, **88**, 2547-2553.
S16. G. te Velde and E. J. Baerends, *J. Comput. Phys.*, 1992, **99**, 84-98.