Diving into chemical bonding: an immersive analysis of the electron charge rearrangement through virtual reality

Andrea Salvadori,[a †] Marco Fusè,[a †] Giordano Mancini,[a *] Sergio Rampino,[a *] and Vincenzo Barone[a]

January 29, 2021

Abstract

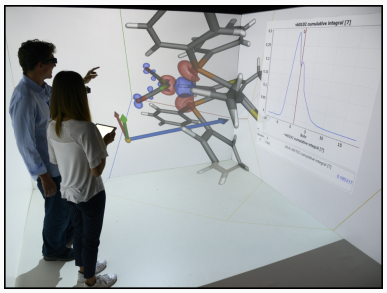
An integrated environment for the analysis of chemical bonding based on immersive virtual reality is presented. By employing a multi-screen stereoscopic projection system, researchers are cast into the world of atoms and molecules, where they can visualize at a human scale the electron charge rearrangement (computed via state-of-the-art quantum-chemical methods) occurring upon bond formation throughout the molecular region. Thanks to specifically designed features, such a virtual laboratory couples the immediacy of an immersive experience with a powerful, recently developed method yielding quantitative, spatially-detailed pictures of the several charge flows involved in the formation of a chemical bond. By means of two case studies on organometallic complexes, we show how familiar concepts in coordination chemistry, such as donation and back-donation charge flows, can be effectively identified and quantified to predict experimental observables.

Keywords: Chemical bonding, Electron density, Charge rearrangement, Molecular visualization, Immersive virtual reality.

[a]SMART Laboratory, Scuola Normale Superiore, Piazza dei Cavalieri 7, 56126 Pisa, Italy

[†]A. Salvadori and M. Fusè contributed equally to this work.

[*]Correspondence to: G. Mancini (giordano.mancini@sns.it) and S. Rampino (sergio.rampino@sns.it).



Based on state-of-the-art virtual-reality technology, we have developed an integrated environment for the immersive analysis of chemical bonding allowing chemists to perceive at a human scale how electrons rearrange upon bond formation and to perform, in an interactive and cooperative way, numerical analysis on the visualized data towards a new paradigm of conducting research in chemistry.

INTRODUCTION

Modern molecular sciences make extensive use of computational methods to predict properties and rationalize experimental outcomes for a great variety of physical systems over a wide range of space and time scales. Due to the recent growth of the available computational power, the results of these theoretical calculations produce data sets of increasing size and complexity. While this situation has had the benefit of encouraging the development of automatic data analysis procedures, it has not diminished the need for the insight ability and reasoning capabilities of the human mind.

Scientific Visualization (SV) approaches this problem by trying to meaningfully represent complex data sets by means of interactive Computer Graphics (CG). The aim is, in fact, to exploit the capability of the human visual system to identify structures, patterns, relations and anomalies in the images for a quicker and deeper understanding of the visualized data. In this respect, recent advances in CG and Immersive Virtual Reality (IVR) technologies have opened unprecedented scenarios whereby users are immersed in three-dimensional (3D) representations of their molecular simulations at a human scale. In this way, IVR allows users to interact naturally with the visualized data and to exploit their proprioceptive system to enhance the perception of the represented system, thus fostering spatial deductions about the dimensions, proportions and topology of complex data such as surfaces, volumetric data and vector fields.

Whereas IVR technology has been so far the preserve of a few laboratories that could afford expensive, highly specialized hardware to assemble room-sized multi-screen projection theaters, such as the Cave Automatic Virtual Environment (CAVE),[1,2] the recent introduction of a new generation of consumer-grade immersive helmets (known as Head-Mounted Displays, HMDs), such as the Oculus Rift[3] or HTC Vive,[4] is rapidly changing this scenario and, thanks to their relatively low cost, will presumably lead in the near future to a wide spreading of IVR technologies also among scientists. However, a critical aspect for the success of this IVR revolution will be the conception of new visualization paradigms able to speed up cognitive processes beyond the simple wonder of being immersed in the nanoscopic world.

Chemistry is itself a deeply visual discipline, its very language being actually based on graphical diagrams. The reader is certainly familiar with 3D ball-and-stick molecular models (with balls representing atoms and sticks signifying chemical bonds). The mathematical modeling underlying these simplified abstractions, on the other hand, is inherently complex (being itself an active research field) and requires appropriate visualization and analysis tools. Indeed, an accurate description of the molecular world involves the computation, by means of rigorous quantum-chemical models, of quantities such as molecular orbitals and electron densities that are formally 3D mathematical functions spanning the entire physical space. As far as chemical bonding is concerned, the pioneering works of Bader on diatoms[5] showed for the first time that a deep insight into the nature of a chemical bond can indeed be gained through a careful analysis of the topology and features of the electron charge rearrangement occurring upon bond formation. In his late-sixties works, Bader made use of 2D contour-line plots analogous to those used for representing reliefs in geographic maps or potential energy surfaces.[6,7] Although merging the benefits of immersive visualization with precise numerical analysis is still a major challenge,[8] nowadays the possibility provided by IVR of interacting with these quantities at a human scale opens unexplored perspectives in the analysis of such data.

Prompted by these ideas and based on the IVR equipment available at the SMART Laboratory of the Scuola Normale Superiore in Pisa, we have developed an integrated environment for the immersive analysis of chemical bonding which couples the immediacy of an IVR experience with interactive analysis tools providing chemical insight on the numerical outcomes of state-of-the-art quantum-chemical calculations.

BOND-ANALYSIS TECHNIQUES

Current theories of chemical bonding originated from Lewis' intuition of electron-pair sharing in his 1916 seminal paper,[9] which received formal justification on the basis of quantum mechanics a few decades later through the Heitler-London treatment of the H_2 molecule[10] and Pauling's work on the chemical bond.[11] One century after Lewis' work, thanks especially to the advent of molecular-orbital (MO) and density functional theory (DFT), several successful

methods for the analysis of chemical bonding based on quantum-chemical calculations have been devised (the reader is referred to Ref. 12 for a comprehensive review). We have recently specialized in a simple, yet powerful method proposed by one of us in collaboration with others – the so-called natural orbital for chemical valence/charge displacement (NOCV/CD) analysis[13,14] – focusing on the changes that the electron density undergoes upon formation a chemical bond.

Given an adduct AB formed by fragments A and B, the electron charge rearrangement taking place after formation of the A−B bond may be formulated as the difference $\Delta\rho(x,y,z)$ between the total electron density of the adduct and a reference electron density, which is associated with the unbound fragments A and B (taken at their in-adduct geometries) and constructed from the occupied molecular orbitals of the isolated fragments previously made orthonormal to each other. As shown in detail in Refs. 15–18, if all densities are worked out from single-determinant wavefunctions (as is the case in Hartee-Fock or DFT calculations), by diagonalizing the so-called 'valence operator'[19–21] and finding its eigenvalues v_k and eigenfunctions φ_k termed as 'natural orbitals for chemical valence' (NOCVs), $\Delta\rho$ can be decomposed into weighed contributions ascribable to pairs of NOCVs coupled by the eigenvalue v_k:

$$\Delta\rho = \sum_k v_k \left(|\varphi_k|^2 - |\varphi_{-k}|^2 \right) = \sum_k v_k \Delta\rho_k \ , \tag{1}$$

where the spatial dependence of densities and orbitals has been dropped for clarity. In other words, the total electron charge rearrangement taking place after the bond formation results from additive charge flows of v_k electrons flowing from the orbital φ_{-k} to the orbital φ_k, with k ranging from one to the number of occupied molecular orbitals of the adduct. Only a few NOCV components in Eq. 1 have a significant weight and thus contribute non-negligibly to the overall charge rearrangement. As will be shown later on in this article, a *qualitative* visual inspection in a 3D space of these few important contributions to the overall charge rearrangement reveals that they have a clear chemical meaning. On the other hand, a *quantitative* estimate of the charge flow along a suitably chosen direction is often desirable and even mandatory if a comparison with experimental data is in order. This can be easily achieved by building, for each of the charge-flow components, the so-called charge-

displacement (CD) function,[13,22] $\Delta q_k(z)$, defined as a progressive partial integration along a suitable axis z of the related electron density difference $\Delta\rho_k$ multiplied by its weight v_k (see Eq. 1), and providing a clear and quantitative picture of the charge flow along a selected direction in space:[23,24]

$$\Delta q_k(z) = v_k \int_{-\infty}^{z} \mathrm{d}z' \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \Delta\rho_k(x,y,z')\ \mathrm{d}x\ \mathrm{d}y \tag{2}$$

with the z axis usually chosen to be the bond axis between the fragments. Accordingly, the CD function at a given point z quantifies the exact amount of electron charge that, upon formation of the bond, is transferred from right to left (the direction of decreasing z) across a plane perpendicular to the bond axis through z.

Molecular orbitals and electron densities are the outcome of electronic-structure calculations routinely carried out with quantum-chemistry packages such as Gaussian, Dalton, ADF and Molpro. Whereas these quantities are formally represented as vectors or matrices reflecting their expansion in a basis set of known functions, for visualization purposes they are commonly discretized onto a finite-volume uniform grid[1] and saved in dedicated file formats (such as gopenmol .plt files and Gaussian Cube files).[25] Unfortunately, most molecular graphics programs provide poor or no support for operating on cube files, and the required numerical data analysis is usually demanded to external utilities such as Gaussian's CUBEMAN, allowing for a few basic operations like add or subtract, or the CUBES suite[26], fully capable of carrying out CD analysis, which however require an additional computational expertise not always part of a chemist's scientific background.

[1]With the term 'uniform grid' we mean a partitioning of a region of the 3D space into equally sized cells. The sampling points are located at the vertices of the cells, and thus are equally spaced along the axes of the grid. Though usually uniform grids are defined on cubic cells, in a general case these can be parallelepipeds. In other words, whereas, as mentioned above, all cells in a uniform grid have the same shape and size, the sides of each cell can have different lengths and may not be orthogonal to each other.

CAFFEINE AND THE VIRTUAL LABORATORY

The benefit of visualizing systems of chemical interests through immersive environments has already been demonstrated[27] and many popular molecular visualizers such as VMD,[28,29] PyMol,[30,31] and Chimera[32] have been adapted to support IVR. The Caffeine project[33,34] was started with the intent of creating a new molecular viewer specifically designed and developed to take the advantage of IVR technologies. Caffeine is conceived to work both on desktops and in CAVE-like IVR facilities and allows the user to visualize both static and dynamic structures in standard representations (all-atoms and ribbons), isosurfaces extracted interactively by volume data sets, and line charts displaying additional scalar data resulting from further data analyses in an augmented-reality fashion. Support for the latest generation of HDMs is under development.

In its latest developments implemented for the present work, Caffeine has been upgraded with computational procedures for carrying out operations on molecular electron densities and orbitals, that as mentioned are routinely computed by popular quantum-chemistry packages and commonly discretized onto a finite-volume uniform grid in dedicated file formats. The required operations for bond analysis that have been implemented in Caffeine range from the basic ones (i.e., linear combinations of 3D grids) to those more complex such as the computation of the CD function over a 3D grid along an arbitrary directional axis z. The reader is referred to Section 3 of the Supporting Information (SI) for a full account of the introduced features.

As already mentioned, we based our virtual laboratory on the CAVE multi-screen projection theater available at the SMART Laboratory of the Scuola Normale Superiore interfaced with the Caffeine software. As will be detailed in the following, researchers interact with the system at hand and with the analysis tools through a remote control application for tablet computers. We note here that even if this work focuses on bond analysis performed in an IVR context, all analysis tools are also available in the desktop version and a step-to-step tutorial is provided in Section 3 of the SI on how to use these tools with the desktop version of Caffeine.

TWO CASE STUDIES OF COORDINATION CHEMISTRY

To illustrate how the developed environment works and show the potential of an IVR-based virtual laboratory for the analysis of chemical bonding, we resort to the rich and complex world of coordination complexes. In this context, a deep comprehension of the several contributions to the electron charge rearrangement taking place after bond formation plays a fundamental role in rationalizing the outcome of experiments and developing new compounds with predetermined characteristics, such as organometallic catalysts capable of driving the outcome and efficiency of catalytic reactions.[35] After giving the relevant computational details, in a first subsection we show the features and the potentialities of the NOCV/CD analysis through the simple case of the metal-carbonyl bond in two complexes, $[CuCO]^+$ and [FCuCO] (the former is also used in the tutorial in the SI). Then, in a second subsection, we show the immersive analysis applied to a real case study on chelation bonding.

Computational details

Geometry optimization and frequency calculations were performed using density functional theory (DFT) with the Gaussian suite of programs (G16 Rev. A.03)[36] adopting the B3LYP[37,38] exchange-correlation functional. Calculations on $[CuCO]^+$ and [FCuCO] were performed in vacuo using a LANL2DZ basis set with effective core potential for copper[39] and a 6-31+G*[40–42] basis set for the other atoms. Calculations on the nickel complexes were performed including the bulk solvent effects by means of the Polarizable Continuum Model (PCM)[43–45] through the integral equation formalism model (IEFPCM) as implemented in Gaussian[46] and using a 6-31G*[40,41,47] basis set for all atoms. In all cases, semiempirical dispersion contributions were taken into account by inclusion of Grimme's D3BJ[48] model as implemented in Gaussian. Anharmonic calculations were performed with the GVPT2 model.[49,50] Cubic and semi-diagonal quartic force constant were computed by numerical differentiation (with displacements of 0.01 Å) of the analytical Hessian along each active normal coordinate. In order to reproduce the anharmonic effects at a reasonable computational cost, a reduced dimensionality (RD) approach[51,52] was adopted where only carbonyl stretching modes were considered as active. Bond analysis was performed by means of the recently

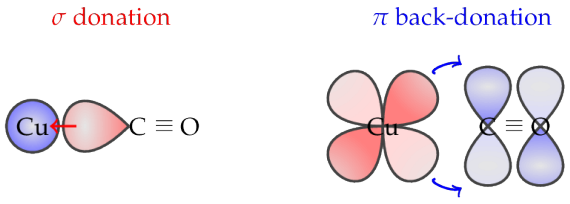


Figure 1: Schematic representation of the σ donation and π back-donation charge flows in the copper-carbonyl coordination bond.

proposed natural orbitals for chemical valence/charge displacement (NOCV/CD) scheme. NOCV analysis was performed by interfacing G16 with an *ad hoc* written program.[53] Charge-displacement analysis on discretized electron densities in the form of Gaussian cube files was carried out with the analysis tools made available in Caffeine.

How NOCV/CD analysis works. The metal-carbonyl bond

The chemical bond between carbon monoxide CO and a transition metal is commonly explained through the synergistic interplay between σ-donation and π-back-donation charge flows (see for example Ref. 54). Considering for instance the prototype case of $[CuCO]^+$, upon coordination of carbon monoxide CO to the copper cation Cu^+, a fraction of the electronic charge from the CO lone pair on the carbon side (the CO highest occupied molecular orbital, HOMO) is expected to flow in the empty orbital of σ symmetry available at the metal center and, at the same time, a fraction of charge from the filled d orbitals of copper having π symmetry is expected to flow in the lowest unoccupied molecular orbital (LUMO) of CO of similar symmetry. These two concurrent charge flows, commonly termed as σ donation and π back-donation, respectively, are sketched in Fig. 1.

To get an accurate and quantitative description of these phenomena, we performed DFT calculations for the $[CuCO]^+$ system and carried out a detailed analysis of the copper-carbonyl bond. The principal results of the NOCV/CD analysis are reported in Table 1 and displayed in Figure 2 while a full account is given in the SI. As apparent from the obtained data, the overall charge rearrangement upon formation of the copper-carbonyl bond results mainly from the NOCV contributions with $k \leq 4$, the others having weight $v_k \leq 0.05$ e and thus contributing negligibly to the overall $\Delta\rho$. A visual inspection of components $k = 1$, 2

and 3 (reported in the left panel of Figure 2) reveals that the most important one ($k = 1$ and weight $v_k = 0.40$) can be identified with the σ donation of the scheme of Figure 1 and is followed by two degenerate components ($v_k = 0.23$) representing the π back-donation. The fourth component which, as detailed in the SI, has a smaller though non negligible weight of 0.17 e, represents only *intra*-fragment charge rearrangement and is omitted here for clarity because it does not contribute to the *inter*-fragment charge transfer that we are interested in.

As mentioned in the previous Section, quantitative estimates of the charge-flow profile associated with each bond component can be obtained by computing the related CD function (Eq. 3). It is worth recalling here that the CD function quantifies the exact amount of electron charge that, upon formation of the bond, is transferred from right to left. Accordingly, negative values of the CD function correspond to charge flow in the opposite direction, i.e. form left to right. CD functions associated with the σ-donation (red line) and the degenerate π-back-donation components (blue line and dots) in the metal-carbonyl bond in $[CuCO]^+$ are shown as full-color curves in the right panel of Figure 2. As evident, the CD curve associated with σ donation is almost always positive in the molecular region, thus indicating a charge flow from right to left, with the only exception of a small portion in the negative z axis in the backside of the metal centre. This behavior can be explained by the related 3D isosurface plots in the left panel of the figure. The charge-rearrangement contribution associated with the NOCV $k = 1$ component results in fact from a flow of 0.40 e from the molecular orbital ϕ_{-1} (displaying amplitude in the carbon lone pair region and

Table 1: Weights (NOCV eigenvalues v_k) and charge-transfer values (CT_k) associated with the σ-donation and π-back-donation components of the charge rearrangement upon formation of the metal-carbonyl bond in $[CuCO]^+$ and [FCuCO].

$[CuCO]^+$				**[FCuCO]**			
k		v_k /e	CT_k /e	k		v_k /e	CT_k /e
1	(σ-don)	0.40	0.16	1	(π_x-back)	0.36	0.11
2	(π_x-back)	0.23	0.04	2	(π_y-back)	0.36	0.11
3	(π_y-back)	0.23	0.04	3	(σ-don)	0.29	0.13

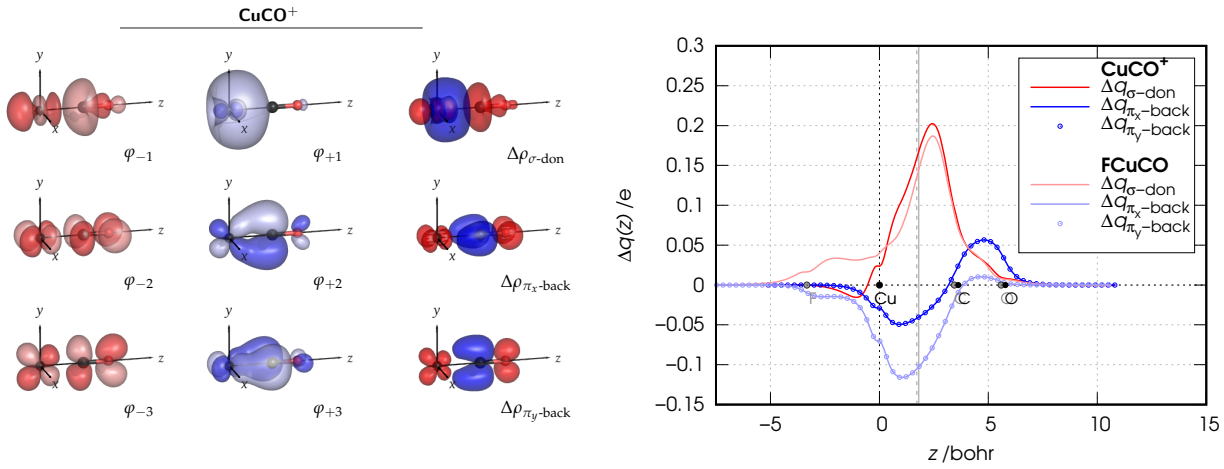


Figure 2: Left panel: NOCV pairs (φ_{-k} and φ_k, isodensity surfaces at ± 0.05 (e/bohr^3)$^{1/2}$) and charge rearrangement ($\Delta\rho$, isodensity surfaces at $\pm$ 0.005 e/bohr^3) associated with the σ-donation ($k = 1$) and π-back-donation ($k = 2, 3$) components of the metal-carbonyl bond in [CuCO]$^+$. In $\Delta\rho$ plots, red isosurfaces represent regions of electron depletion, blue isosurfaces represent regions of electron accumulation. Right panel: CD functions for the σ-donation and π -back-donation components of the metal-carbonyl bond in [CuCO]$^+$ and [FCuCO].

in a d_{z^2}-like region at the metal centre) to the molecular orbital ϕ_{+1} (mainly representing a s orbital of the metal). This means that a *sd* hybridization is taking place at the metal once accepting charge due to the σ donation from CO, and it is precisely to this internal charge rearrangement that the above mentioned negative portion of the CD curve is due. It is interesting to note that only a fraction of the overall rearrangement of 0.40 e is actually transferred from CO to the metal upon bond formation, the remaining part being involved in the *intra*-fragment rearrangement only. A reasonable estimate of the charge transfer (CT) between the two fragments may indeed be obtained by taking the value of the CD function at the middle of the bond (solid gray line in Figure 2). Indeed, according to Table 1, the fraction of electron transferred from right to left across the gray line is only 0.16 e.

As to the degenerate π-back-donation components, the associated CD curves show negative values in the metal-carbon region. As expected, this corresponds to a positive charge

flow in the direction from left to right, i.e. from the metal to the carbonyl group. At the same time, the curve shows a positive peak at the carbon-oxygen bond, thus indicating (as extensively shown in Ref. 24) a polarization of the CO electrons in the C ← O direction due to the positive charge on the copper atom. As appears in Table 1, whereas a total of 0.23 e is involved in each of the two degenerate π-back-donation charge flows, only a fraction of 0.04 e is actually being transferred from the metal to the carbonyl moiety.

The overall picture radically changes if a fluorine ion is added as a ligand to copper and the complex [FCuCO] is considered. As the reader may easily get from the related CD functions shown as light-color curves in Figure 2, the absence of a positive charge at the metal centre enhances the π back-donation and reduces the C ← O polarization of the π CO orbitals. Also, the σ-donation component appears affected, thus featuring a smaller amount of *inter*-fragment charge transfer and a larger spatial extension of the charge flow donated from the carbonyl group which reaches far into the backside of fluorine.

Immersive bond analysis applied to a real case study

The simple cases reported in the previous section nicely illustrate the amount and quality of information that can be extracted from the NOCV/CD analysis of a chemical bond. The relative simplicity of the data involved made it feasible to get satisfactory results based on 2D representations. Indeed, real research studies may involve much more complicated molecular systems and the assignment of a chemical character to charge-flow components upon bond formation may become a tough and error-prone task if based on whatever nice isosurface plots such as those of Figure 2. However, the possibility of an immersion into these quantities by exploring their topological features in a full 3D space through IVR can enormously simplify this task and allows for gaining perspectives which would remain unexplored otherwise.

In a recent study[55] we investigated the detailed metal-ligand bonding features in the nickel dicarbonyl complexes of general formula $[Ni(CO)_2(PP)]$, with PP being one of the atropoisomeric chelating diphsophine ligands in the top row of Figure 3. It was thanks to a number of IVR sessions that we could find the key to the understanding of the amount of computational data produced.

A typical session in our virtual laboratory starts with a preparatory stage where the

required files (volumetric data sets and geometries) are loaded into Caffeine from a desktop-based control panel outside the CAVE (additional scalar grids can be derived as linear combination of the previously loaded/computed ones). Then researchers, equipped with active glasses and a tracking sensor, enter the CAVE room and interact in a cooperative fashion with the 3D representations of the loaded volumetric data through a remote control application for tablet computers.[2] The remote control allows the user to move, rotate and scale the displayed molecular system. A second panel contains the controls related to the directional analysis described in the previous section. When activated, a special reference frame is shown in the 3D scene, whose origin and z axis are used as parameters for the CD function. Users can therefore position and orientate the representation of the molecular structure and of the related electron charge rearrangement until an optimal point of view of the system under investigation is obtained as well as can physically move themselves within the CAVE. Then, they can activate the analysis mode to move and rotate the analysis reference frame according to their needs. By pressing a dedicated button of the remote control, the CD function is evaluated for each point along the chosen axis, and the result is plotted on a line chart displayed within the 3D scene in an augmented-reality fashion as if it were the outcome of an experimental measurement (see Figure 4 for a real picture portraying two users in an IVR session in our virtual laboratory). By default the chart is always in front of the tracked user and drawn on a semi-transparent quad. However, the user can in any time 'pin' the chart in a particular position, switch to a different chart or hide it. Finally, a special tool, denoted as 'marking plane' is provided to further support the numerical analysis of the CD function. The 'marking plane' is a plane orthogonal to the chosen analysis direction and can be moved along it. Its purpose is to query the value of the CD function for particular points of interest: once the 'marking plane' has been placed at the desired position, it is sufficient to press a dedicated button on the remote control to insert a corresponding vertical marker on the currently displayed chart and get the associated

[2]It is important to note that even if, strictly speaking, CAVEs are usually single-user systems (in the sense that the images are generated by taking into account the position and the orientation of the head of the user wearing the tracking markers), small groups of people (2-3) can participate to the IVR session with satisfactory results as long as they stay close enough to the tracked user and watch approximately in the same direction.

<x,y> value pair displayed in a dedicated table placed below the chart. This process can be repeated, so as to mark the plotted CD function in several points of interest.

After inspection of the main charge-flow contributions to the overall charge rearrangement upon the metal-ligand bond formation, we could isolate the charge-flow component associated with the σ donation from the phosphorus lone pairs of the chelating diphosphane and resulting from the additive contribution of NOCV components with $k = 1$ and $k = 2$ (see a sketch of the resulting component in the bottom left panel of Figure 3). We could further demonstrate that the amount of charge transferred due to such charge-flow component is in strict correlation with a structural property of the complex itself that can be measured experimentally, namely the symmetric carbonyl stretching frequency ν_{CO} (sym). In other words, besides proving the robustness of the NOCV/CD analysis scheme, we provided the framework in which spectroscopic data on coordinated CO may be used to predict electronic properties of the ligands such as, in the case at hand, the σ-donor ability (see Ref. 56 for a scheme, based on these analysis tools, to switch the carbonyl stretching frequency into a selective probe of the π-acceptor ability).

In the bottom right panel of Figure 3 we report newly calculated correlations of CTs with ν_{CO} (sym), obtained through the developed machinery as follows. Red circles are CTs resulting from a CD analysis along the z axis of the red reference frame (as sketched in the bottom left panel of Figure 3) connecting Ni with one chelating P and determined by placing the 'marking plane' at the middle of the Ni-P bond. Blue circles are CTs resulting from a CD analysis along the z axis of the blue reference frame connecting Ni with the other chelating P and determined by placing the 'marking plane' at the middle of the Ni-P bond. Black circles are CTs extracted from a CD analysis along the black z axis (bisecting the $\widehat{(P)Ni(P)}$ angle and lying on the $Ni(P)_2$ plane) and derived by placing the 'marking plane' at the average z coordinate of the two P's. The R^2 correlation values reported in the plot (0.663, 0.789 and 0.988 for the red, blue and black circles, respectively) clearly show that it is the cooperative effect of each phosphorus-to-nickel donation that drives the CO stretching in the complexes and that a poorer correlation is obtained when the donation of only one P at a time is considered.

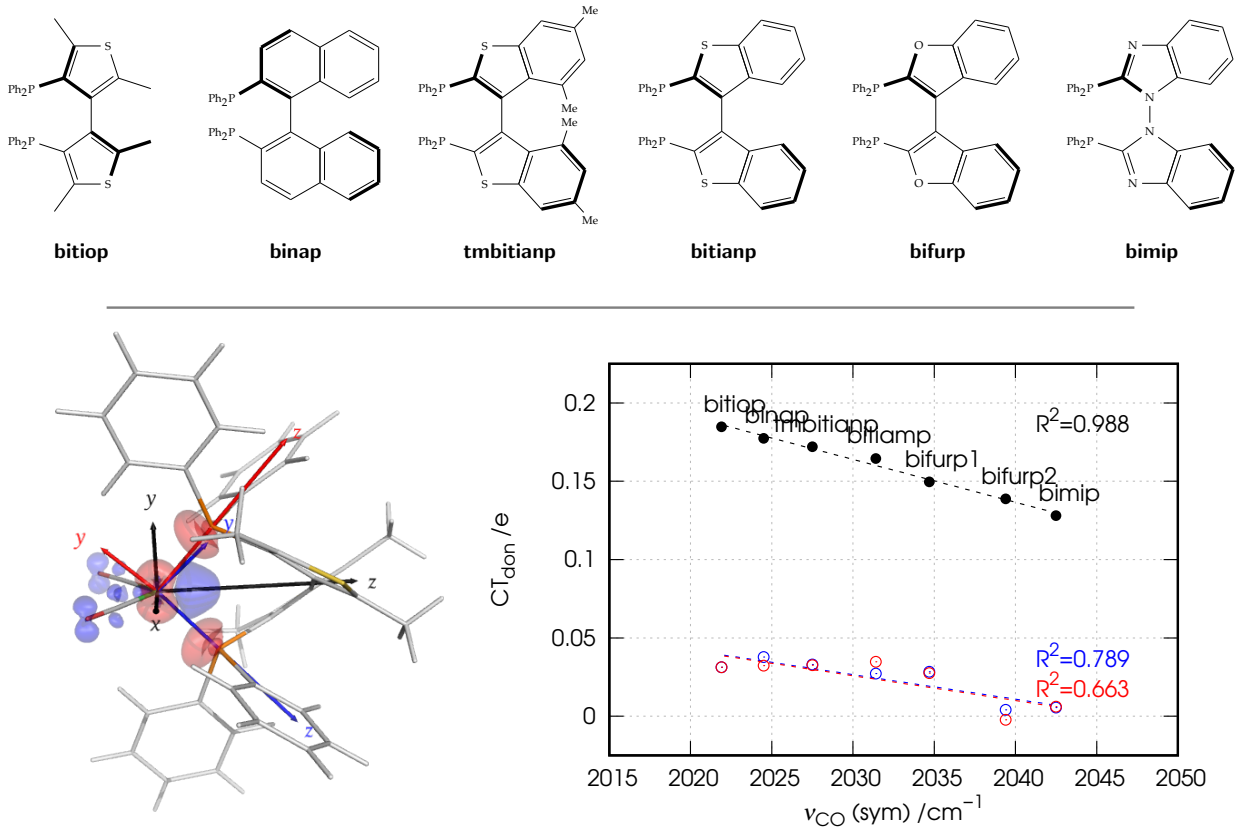


Figure 3: Top row: chemical structures and abbreviated names for the six atropoisomeric chelating diphosphane ligands, PP, in considered series of nickel dicarbonyl complexes [Ni(CO)$_2$(PP)]. Bottom, left panel: charge rearrangement ($\Delta\rho$, isodensity surfaces at $\pm$ 0.006 e/bohr^3) associated with the σ-donation (resulting from the components $k = 1$ and $k = 2$) of the metal-carbonyl bond in the Ni(CO)$_2$(bitiop) complex. Red isosurfaces represent regions of electron depletion, blue isosurfaces represent regions of electron accumulation. Bottom, right panel: Correlation between the symmetric carbonyl stretching frequencies ν_{CO} (sym) and the charge transfer ($\mathrm{CT_{don}} = \mathrm{CT_1} + \mathrm{CT_2}$) associated with the ligand-to-metal donation from the phosphorus lone pairs of the ligands. Labels bifurp1 and bifurp2 refer to two conformers of the bifurp complex of comparable energy.

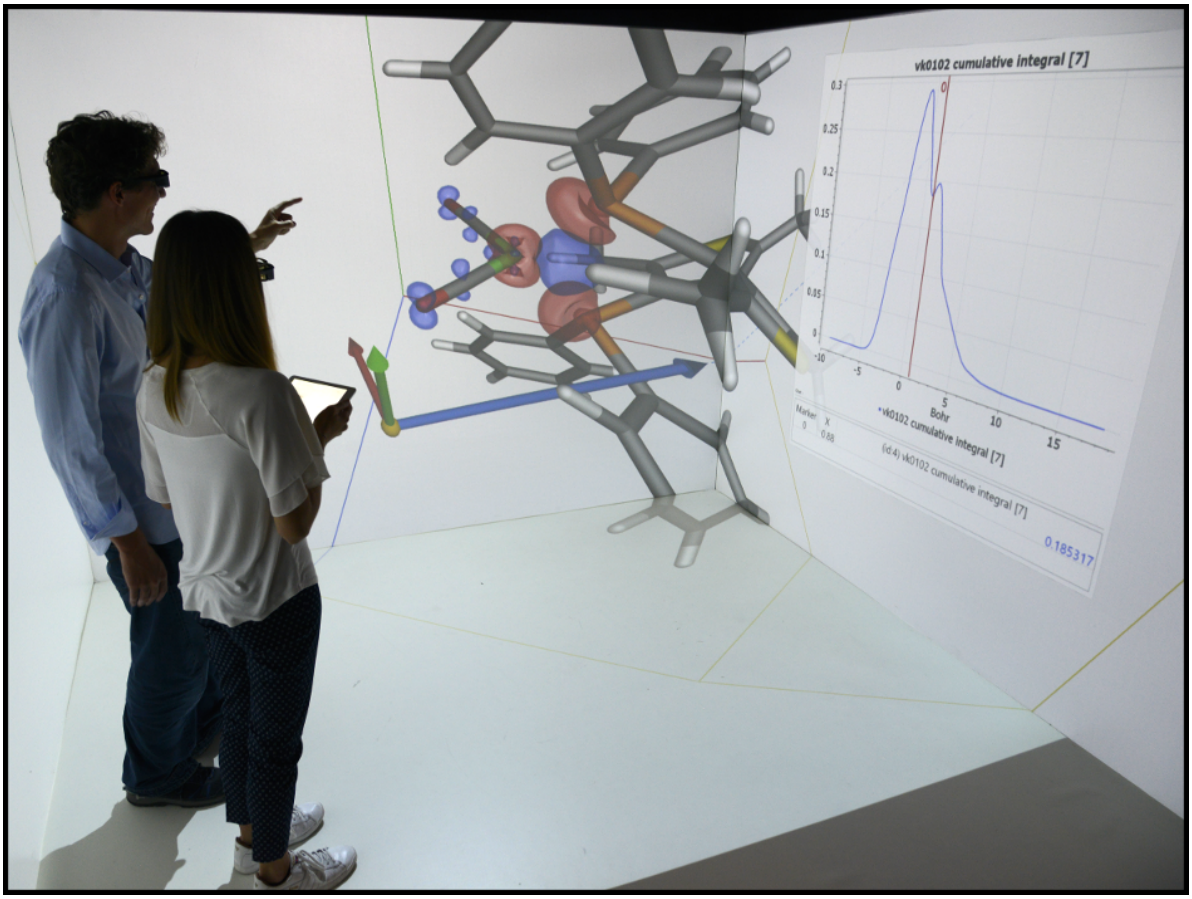


Figure 4: Two users in the virtual laboratory while analyzing the chemical bond between nickel and a chelating disphosphane in one of the nickel dicarbonyl complexes considered in this work. Researchers interact with the electron charge rearrangement at hand (red lobes: depletion, blue lobes: accumulation) and with the analysis tool (represented by a reference frame for the directional analysis discussed in the article) through a remote control application for tablet computers. For the sake of clarity, the stereo mode of projectors was temporarily disabled to shoot this photo.

PERSPECTIVES

Modern IVR technologies are increasingly used in the world of academic research in chemistry, thus offering a novel, unprecedented perspective for representing and analyzing the realm of atoms and molecules, and thus demanding a radical change in the paradigms adopted so far to this purpose. It is in this context that we have conceived and developed an IVR-based virtual laboratory for the analysis of chemical bonding. This casts researchers into the molecular world and provides them with interactive analysis tools for studying the detailed features of chemical bonding based on accurate theoretical calculations, in a way that resembles much closely that of experimentalists in a laboratory experiment (the reader is referred to the Caffeine Web page (http://smart.sns.it/caffeine) where video recordings of IVR sessions in the virtual laboratory can be accessed and instructions on how to get the Caffeine software are made available). The potentialities of our virtual laboratory have been illustrated through two case studies taken from coordination chemistry, thus providing a thorough, quantitative analysis of the detailed features of the metal-carbonyl bonding. Clearly, the applicability of the developed machinery goes well beyond coordination chemistry, naturally extending to all kinds of chemical interactions and also to the analysis of the electron charge rearrangement upon electron excitation as well as to the representation of condensed-phase processes. From a technical point of view, there are virtually no limitations to the size of the system to be analyzed as these are largely preceded by limitations in solving the related electronic-structure problem. Moreover, the analysis can be performed on the charge rearrangement computed by the most popular quantum-chemistry programs as far as it is provided to the virtual laboratory in the widely supported .cube file format (or converted to .cube file from other similar formats).

We believe that the development of virtual laboratories, like that presented in this article, will pave the way for a revolutionary paradigm of teaching and doing research in chemistry. This would allow for the inherent complexity of a theoretical and computational treatment to become increasingly transparent to scientists, with the old-style keyboard & terminal approaches to the interaction and analysis of data being replaced by more 'natural' ones. To pursue this aim, our group is actively working on improving the user interaction by replacing

the tablet with handheld VR controllers and designing an internal 3D graphical user interface (GUI), thus allowing the user to place, scale and orient the molecular system and to handle the analysis tools via hand-driven gestures. Another crucial feature we are working on is the support for latest generation of consumer-grade HMDs, so as to allow individuals and small research groups to take advantage of IVR for educational and research purposes.

ACKNOWLEDGMENTS

The authors are grateful to Cristina Puzzarini (Università di Bologna) for careful reading of the manuscript and valuable suggestions for improving its readability, and to the technical staff of the SMART Laboratory for managing the computational facilities at SNS.

((Additional Supporting Information may be found in the online version of this article.))

APPENDIX: DETAILS ON THE ALGORITHM FOR INTEGRATION ALONG AN ARBITRARY DIRECTIONAL AXIS

The main issue in performing the integration of Eq. 2 in the article is that the 3D function to be integrated is discretized as 'volume data' onto a finite-size 3D grid. This means that, as is common, if the integration axes and steps are not the same of the mentioned grid, a 3D interpolation is in order. We discuss here the algorithm to perform the integration of Eq. 2 focusing on the calculation of the following quantity:

$$\Delta\rho(z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \Delta\rho(x,y,z)\ \mathrm{d}x\ \mathrm{d}y\ , \tag{3}$$

the remaining cumulative integration over the z variable being a trivial task on a 1D function.

In the following, the relevant steps of the algorithm will be discussed. For sake of clarity, we will make use of a simplified source code written in a "pseudo-C++" programming language, in which the details which are not relevant for this discussion have been omitted.

```
1 vector<pair<double, double>> integrateAlongDirectionalAxis(Grid3D
    grid, vec3 P_intg, vec3 Z_intg)
2 {
```

```
3       vector<pair<double, double>> result;
```

The signature of the function is shown above. It takes three parameters as input: a 3D uniform grid representing the volumetric data set (grid), the origin of the analysis reference frame (P_{intg}), and the directional axis ($\vec{Z}_{\text{intg}}$). Substantially, the algorithm "marches" along $\vec{Z}_{\text{intg}}$ with steps of fixed size. For each point P_Z encountered during the march, the integral over the plane orthogonal to $\vec{Z}_{\text{intg}}$ and passing through P_Z is computed. The function returns a vector (dynamic array) of pairs, in which the first element of the pair is the distance between P_Z and P_{intg}, while the second element is the value of the integral computed over the related plane. The only precondition is that P_{intg} must lie within the bounds of the grid.

```
4       vec3 X_intg, Y_intg = computeOrthogonalBasis(Z_intg);
5       double delta = 0.5 * min( grid.cellLengthX(), grid.cellLengthY
        (), grid.cellLengthZ() );
```

The first step of the algorithm consists in computing two versors, named $\vec{X}_{\text{intg}}$ and $\vec{Y}_{\text{intg}}$, such that the triple $\{\vec{X}_{\text{intg}}, \vec{Y}_{\text{intg}}, \vec{Z}_{\text{intg}}\}$ is an orthogonal basis. $\vec{X}_{\text{intg}}$ and $\vec{Y}_{\text{intg}}$ represent the integration directions over the plane. The steps of the march along $\vec{Z}_{\text{intg}}$, as well as the integration steps along $\vec{X}_{\text{intg}}$ and $\vec{Y}_{\text{intg}}$, have a fixed size equal to the half of the smallest side of the cell of the grid.

```
6       Parallelepiped bounds = computeBounds(grid);
```

Although in theory the integral should be computed in the range $[-\infty, +\infty]$ along $\vec{X}_{\text{intg}}$ and $\vec{Y}_{\text{intg}}$, in practice we can sample the value of the 3D function only within the bounds of the grid. This is not a problem as long as the 3D function tends to zero in proximity and over the bounds. However, we must compute proper integration ranges by taking into account the bounds of the grid. In the general case, the shape of the cells of the grid, and therefore of the whole grid, is a parallelepiped. Therefore, we store the bounds of the grid in a data structure designed to represent a parallelepiped.

```
7       double t_Z^min = MAX_DOUBLE;
8       double t_Z^max = LOWEST_DOUBLE;
9       for(int i = 0; i < 8; i++)
```

```
10      {
11          double t = dot(Z_intg, bounds.vertices[i]-P_intg);
12          t_Z^min = min(t_Z^min, t);
13          t_Z^max = max(t_Z^max, t);
14      }
15      int i_Z^min = ceil(t_Z^min/delta);
16      int i_Z^max = floor(t_Z^max/delta);
```

The range to be used in the march along $\vec{Z}_{\text{intg}}$ is computed by taking the minimum and the maximum scalar values ($t_Z^{\min}$ and $t_Z^{\max}$) resulting from the projection of the vertices of the parallelepiped on $\vec{Z}_{\text{intg}}$. To simplify the implementation of the iteration over this range, $t_Z^{\min}$ and $t_Z^{\max}$ are converted in integer multiples of the step size ("delta"): $i_Z^{\min}$ and $i_Z^{\max}$. Note that we round up $i_Z^{\min}$ to the next integer and instead round down $i_Z^{\max}$ to the previous integer, so to be sure to remain inside the bounds of the grid.

```
17      for(int iterZ = i_Z^min; iterZ <= i_Z^max; ++iterZ)
18      {
19          vec3 P_Z = P_intg + (Z_intg * iterZ * delta);
20          double valuesSum = 0.0;
```

The march along $\vec{Z}_{\text{intg}}$ is implemented as a "*for*" loop from $i_Z^{\min}$ to $i_Z^{\max}$. Each iteration begins by computing the coordinates of the corresponding P_Z. We also declare (and initialize to zero) the variable "valuesSum", in which we will store the sum of the values of the volumetric data set sampled over the plane orthogonal to $\vec{Z}_{\text{intg}}$ and passing through the current P_Z.

```
21          double t_Y^min, t_Y^max =  bounds.intersectsLine(P_Z, Y_intg);
22          int i_Y^min = ceil(t_Y^min/delta);
23          int i_Y^max = floor(t_Y^max/delta);
24          for(int iterY = i_Y^min; iterY <= i_Y^max; ++iterY)
25          {
26              vec3 P_Y = P_Z + (Y_intg * iterY * delta);
27              double t_X^min, t_X^max =  bounds.intersectsLine(P_Y, X_intg);
```

```
28              int i_X^min = ceil(t_X^min/delta);
29              int i_X^max = floor(t_X^max/delta);
30              for(int iterX = i_X^min; iterX <= i_X^max; ++iterX)
31              {
32                  vec3 P_X = P_Y + (X⃗_intg * iterX * delta);
33                  valuesSum += grid.sampleValue(P_X);
34              } // End of the march along X⃗_intg
35          } // End of the march along Y⃗_intg
```

The listing above describes the marching procedure over the plane. Starting from P_Z, we perform a march over $\vec{Y}_{\text{intg}}$ where, at each step, we compute the coordinates of a point P_Y. Such point will then be used in turn as the starting point of the march along $\vec{X}_{\text{intg}}$. In terms of code, this corresponds to two nested loops. For the reasons explained above, we need to find proper ranges to iterate along $\vec{Y}_{\text{intg}}$ and $\vec{X}_{\text{intg}}$. In particular, these ranges are obtained by computing the intersection between the parallelepiped and (i) the line directed along $\vec{Y}_{\text{intg}}$ passing through P_Z and (ii) the line directed along $\vec{X}_{\text{intg}}$ passing through P_Y. To this purpose, we implemented the intersection algorithm described in.[57] Again, we express these ranges as integer multiples of the step size ("delta"). Each iteration along $\vec{X}_{\text{intg}}$ corresponds to a point P_X at which we sample the volume by means of a trilinear interpolation. During the march along the plane, we incrementally sum the sampled values in the variable "valuesSum".

```
36          double deltaArea = delta * delta;
37          double integral = deltaArea * valuesSum;
38          double t_PZ = delta * iterZ;
39          result.append(t_PZ, integral);
40      } // End of the march along Z⃗_intg
41      return result;
42  }
```

Once the march along the plane is over, it is easy to obtain the integral of the 3D function over the plane by multiplying the sum of the sampled values by the sampling area. Finally, P_Z is expressed as the signed distance from P_{intg}, and inserted in pair with the value of the

related integral in the vector of the results. The function ends with the end of the march along $\vec{Z}_{\text{intg}}$, by returning the vector of the results.

The complete pseudo code of the function is reported in Listing 1 at the end of this subsection.

In order to maximize the performance of the algorithm, and consequently to reduce the waiting times, we parallelized the computation of the integral of a plane (lines 24-35 in the above code listing) by means of the OpenMP technology (`http://www.openmp.org/`). By doing so, we speed up the algorithm up to a factor of about 5X on our machine equipped with 8 logical processors. Table 2 lists the average completion times, and the related standard deviation, of the algorithm described in this subsection obtained when varying the number of OpenMP threads. It is important to note that, since the areas on which the integral is computed during the march along the directional axis depends on the choice of the directional axis, the workload assigned to each thread of the available pool changes too. For this reason, we performed the benchmark for two representative cases of the directional axis: a cardinal axis (uniform integration area along the march) and an arbitrary one (varying integration area along the march). The benchmarks has been performed on a machine equipped with 32GB of RAM and an Intel Core i7 6700 CPU featuring 4 physical cores and Hyper-Threading technology, for a total of 8 logical processors. The employed operating system is Windows 10 Enterprise 64 bit. The program has been compiled with the Microsoft Visual C++ 14 compiler, supporting OpenMP version 2.0. We found that, on this platform, the "dynamic" scheduling policy is the one that provides the best performance. It can be noticed that increasing the number of OpenMP threads over the number of logical processors does not have a significant impact on the completion time. In fact, as you can observe in Figure 5, the average value and dispersion of the completion times obtained when increasing the number of threads over the number of logical processors is very similar. The reason is that, by using the dynamic scheduling policy with a default "chunk size" of 1, the scheduler initially assign the execution of an iteration of the loop to a thread of the pool, up to reach the number of logical processors. Then, whenever a thread finishes its job, the scheduler dynamically assign one of the remaining iterations to one of the awaiting threads. Therefore, if the number of the threads in the pool exceeds the number of logical processors, those in excess

will simply remain unused. Although during these tests we manually set the number of OpenMP threads used by the program, the average user does not have to manually tune the OpenMP parameters to obtain good performance: OpenMP by default employs a number of threads equals to the number of the available logical processors, so to provide very good performance without the need of an explicit configuration.

	Directional axis: X		Directional axis: $[1,1,1] \cdot (1/\sqrt{3})$	
Num. OMP threads	Average completion time (ms)	Standard deviation (ms)	Average completion time (ms)	Standard deviation (ms)
1	**8566.6**	**72.43**	**9107.3**	**63.26**
2	4130.3	21.89	4340.1	25.91
4	2118.2	56.56	2280.0	49.17
6	1951.7	26.42	2106.2	28.86
8	**1727.7**	**16.61**	**1854.0**	**18.37**
10	1722.9	15.67	1853.5	15.33
12	1724.7	26.90	1849.8	15.32
14	1715.6	14.04	1846.3	17.34

Table 2: Average completion times and related standard deviation of the algorithm of Listing 1, obtained when varying the number of OpenMP threads and for different directional axes. For each examined directional axis and number of threads, ten measurements of the completion time have been performed, on the base of which the average times and standard deviations have been computed. The 3D uniform grid used as input for the algorithm is composed by 283 x 177 x 147 sampled points ("voxels"). The results highlighted in red has been obtained by disabling the OpenMP support when compiling the program, so to evaluate the serial version of the algorithm. The results highlighted in blue, instead, been obtained using OpenMP with a thread pool equals to the number of logical processors. For all the measurements apart from the first, we employed a "dynamic" scheduling policy with a default "chunk size" of 1. The program has been compiled with the Microsoft Visual C++ 14 compiler, supporting OpenMP version 2.0. The "/O2" compilation flag has been used so to optimize the executable for maximum speed. The benchmarks has been performed on a machine running Windows 10 Enterprise 64 bit, equipped with 32GB of RAM and an Intel Core i7 6700 CPU featuring 4 physical cores with a dynamic clock frequency ranging from 3.4 to 4 GHz. Thanks to the Hyper-Threading technology, it exposes to the operating system 8 logical processors.

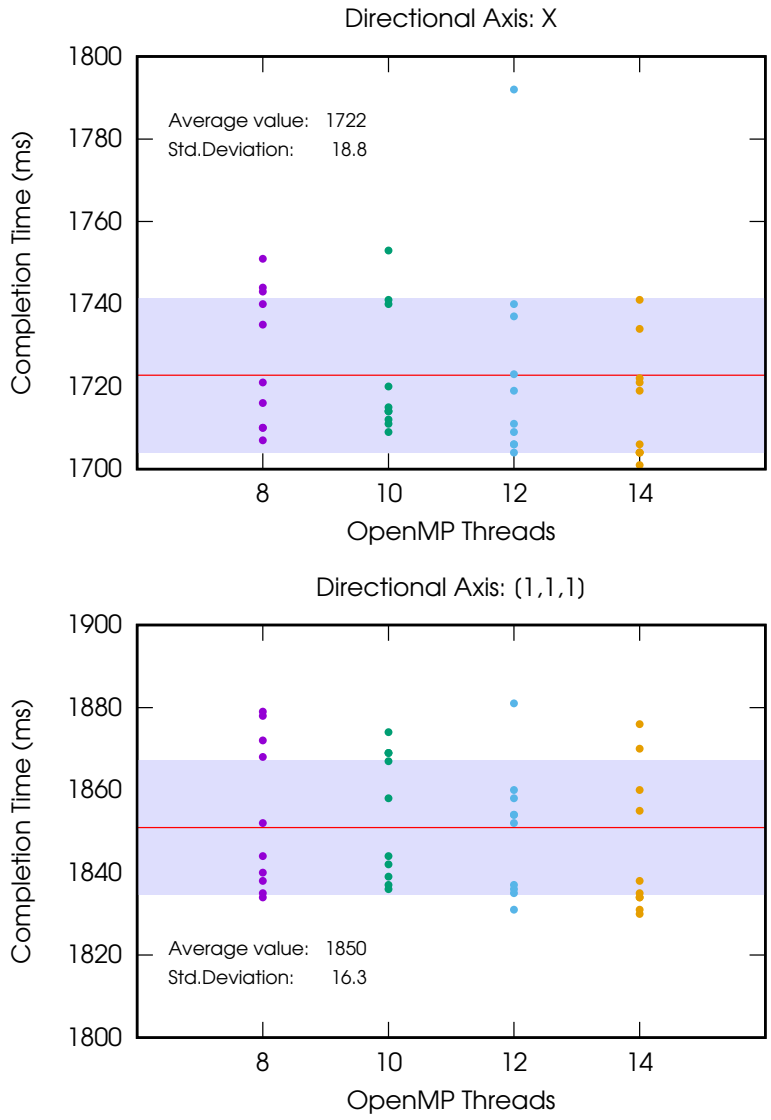


Figure 5: Dispersion of the completion times of the algorithm of Listing 1 with a number of OpenMP threads equals to or greater the number of logical processors (8). For each examined directional axis and number of threads, ten measurements of the completion time have been performed (showed in the charts as points. The overall average values and standard deviations are reported in the figures, and graphically represented respectively by a red horizontal marker at the center of a gray region.

Listing 1: "Pseudo-code of the algorithm to perform the integration of Eq.2 in the article."

```
/*
 * PARAMETERS:
 * grid - The volumetric dataset.
 * P_intg - Origin of the analysis reference frame.
 * Z_intg - The directional axis.
 * PRECONDITION: The starting point must lie within the bounds of the grid.
 */
vector<pair<double, double>> integrateAlongDirectionalAxis(Grid3D
    grid, vec3 P_intg, vec3 Z_intg)
{
    vector<pair<double, double>> result;

    // Given the directional axis, computes two versors so to
    obtain an orthogonal basis.
    vec3 X_intg, Y_intg = computeOrthogonalBasis(Z_intg);

    // Integration step size: half of the smallest side of the
    cells.
    double delta = 0.5 * min( grid.cellLengthX(), grid.cellLengthY
    (), grid.cellLengthZ() );

    // Computes the bounding volume of the 3D grid.
    Parallelepiped bounds = computeBounds(grid);

    // The iteration interval along Z_intg is computed by taking the
    minimum
    // and the maximum scalar values resulting from the projection
    of the vertices of the
    // bounding volume on the directional axis.
```

```
double t_Z^min = MAX_DOUBLE;
double t_Z^max = LOWEST_DOUBLE;
for(int i = 0; i < 8; i++)
{
    double t = dot(Z_intg, bounds.vertices[i]-P_intg);
    t_Z^min = min(t_Z^min, t);
    t_Z^max = max(t_Z^max, t);
}

// Express t_Z^min and t_Z^max as multiples of delta.
// The result is rounded up for i_Z^min and rounded down for i_Z^max,
// so to be sure to remain inside the grid.
int i_Z^min = ceil(t_Z^min/delta);
int i_Z^max = floor(t_Z^max/delta);

// Iterates along Z_intg
for(int iterZ = i_Z^min; iterZ <= i_Z^max; ++iterZ)
{
    vec3 P_Z = P_intg + (Z_intg * iterZ * delta);

    // Stores the sum of the volume values sampled from
    // the X_intg Y_intg plane passing through P_Z.
    double valuesSum = 0.0;

    // The integration interval along Y_intg is obtained by
computing the
    // intersection between the bounding volume of the grid
and the line
    // directed along Y_intg and passing through P_Z.
    // The results are then expressed as multiples of delta
```

```
and rounded.
    double t_Y^min, t_Y^max = bounds.intersectsLine(P_Z, Ŷ_intg);
    int i_Y^min = ceil(t_Y^min/delta);
    int i_Y^max = floor(t_Y^max/delta);

    // Iterates along Ŷ_intg
    for(int iterY = i_Y^min; iterY <= i_Y^max; ++iterY)
    {
        vec3 P_Y = P_Z + (Ŷ_intg * iterY * delta);

        // The integration interval along X̂_intg is obtained by
computing the
        // intersection between the bounding volume of the
grid and the line
        // directed along X̂_intg and passing through P_Y.
        // The results are then expressed as multiples of
delta and rounded.
        double t_X^min, t_X^max = bounds.intersectsLine(P_Y, X̂_intg);
        int i_X^min = ceil(t_X^min/delta);
        int i_X^max = floor(t_X^max/delta);

        // Iterates along X̂_intg
        for(int iterX = i_X^min; iterX <= i_X^max; ++iterX)
        {
          vec3 P_X = P_Y + (X̂_intg * iterX * delta);

            // Sum of the values sampled from the grid in
correspondence of P_X.
            // The sampling is computed by linearly
interpolating the values mapped
```

```
                // to the vertices of the cell containing the
    sampling point P_X.
                valuesSum += grid.sampleValue(P_X);
            } // End of the march along X_intg
        } // End of the march along Y_intg

        // Computes the value of the integral along the X_intg Y_intg
    plane passing through P_Z.
        double deltaArea = delta * delta;
        double integral = deltaArea * valuesSum;

        // Stores P_Z and the corresponding value of the integral
    computed above
        // in a vector (dynamic array). Note that, for this
    purpose, P_Z is expressed
        // as the distance between P_intg and P_Z.
        double t_PZ = delta * iterZ;
        result.append(t_PZ, integral);
    } // End of the march along Z_intg

    return result;
}
```

References

1. C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart, Commun. ACM **35**, 64 (1992).

2. C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti, in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (ACM, New York, NY, USA, 1993), SIGGRAPH '93, pp. 135–142.

3. Oculus VR LLC, *Oculus Rift*, https://www.oculus.com/rift/, (Accessed April 11, 2018).

4. HTC Corporation, *Vive*, https://www.vive.com, (Accessed April 11, 2018).

5. R. F. W. Bader, W. H. Henneker, and P. E. Cade, J. Chem. Phys. **46**, 3341 (1967).

6. S. Rampino, D. Skouteris, and A. Laganà, Theor. Chem. Acc. **123**, 249 (2009).

7. S. Rampino, D. Skouteris, and A. Laganà, Int. J. Quantum Chem. **110**, 358 (2010).

8. J. D. Hirst, D. R. Glowacki, and M. Baaden, Faraday Discuss. **169**, 9 (2014).

9. G. N. Lewis, J. Am. Chem. Soc. **38**, 762 (1916).

10. W. Heitler and F. London, Z. Phys. **44**, 455 (1927).

11. L. Pauling, *The nature of the chemical bond* (Cornell University Press, Ithaca, 1939).

12. G. Frenking and S. Shaik, eds., *The Chemical Bond* (Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, 2014).

13. G. Bistoni, S. Rampino, F. Tarantelli, and L. Belpassi, J. Chem. Phys. **142**, 084112 (2015).

14. M. De Santis, S. Rampino, H. M. Quiney, L. Belpassi, and L. Storchi, J. Chem. Theory Comput. **14**, 1286 (2018).

15. M. Mitoraj and A. Michalak, J. Mol. Model. **13**, 347 (2007).

16. M. Radoń, Theor. Chem. Acc. **120**, 337 (2008).

17. A. Michalak, M. Mitoraj, and T. Ziegler, J. Phys. Chem. A **112**, 1933 (2008).

18. M. P. Mitoraj, A. Michalak, and T. Ziegler, J. Chem. Theory Comput. **5**, 962 (2009).

19. R. F. Nalewajski, A. M. Köster, and K. Jug, Theor. Chim. Acta. **85**, 463 (1993).

20. R. F. Nalewajski and J. Mrozek, Int. J. Quantum. Chem. **51**, 187 (1994).

21. R. F. Nalewajski, J. Mrozek, and A. Michalak, Int. J. Quantum Chem. **61**, 589 (1997).

22. L. Belpassi, I. Infante, F. Tarantelli, and L. Visscher, J. Am. Chem. Soc. **130**, 1048 (2008).

23. S. Rampino, L. Storchi, and L. Belpassi, J. Chem. Phys. **143**, 024307 (2015).

24. G. Bistoni, S. Rampino, N. Scafuri, G. Ciancaleoni, D. Zuccaccia, L. Belpassi, and F. Tarantelli, Chem. Sci. **7**, 1174 (2016).

25. Gaussian Inc., *The cubegen utility*, http://gaussian.com/cubegen/, (Accessed April 11, 2018).

26. S. Rampino, VIRT&L-COMM **7**, 6 (2015).

27. E. Moritz and J. Meyer, in *Proceedings. Fourth IEEE Symposium on Bioinformatics and Bioengineering* (2004), pp. 503–507.

28. W. Humphrey, A. Dalke, and K. Schulten, J. Mol. Graphics. **14**, 33 (1996).

29. J. E. Stone, A. Kohlmeyer, K. L. Vandivort, and K. Schulten, in *Advances in Visual Computing: 6th International Symposium, ISVC 2010, Las Vegas, NV, USA, November 29 – December 1, 2010, Proceedings, Part II*, edited by G. Bebis, R. Boyle, B. Parvin, D. Koracin, R. Chung, R. Hammound, M. Hussain, T. Kar-Han, R. Crawfis, D. Thalmann, et al. (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010), pp. 382–393.

30. Schrödinger LLC., *The PyMOL Molecular Graphics System, Version 1.8*, http://www.pymol.org/, (Accessed April 11, 2018).

31. Virtalis Inc., *VR For PyMOL*, http://www.virtalis.com/vr-for-pymol, (Accessed April 11, 2018).

32. T. D. Goddard, C. C. Huang, E. C. Meng, E. F. Pettersen, G. S. Couch, J. H. Morris, and T. E. Ferrin, Protein Sci. (2017), published online September 6, 2017. DOI: 10.1002/pro.3235.

33. SMART Laboratory, *Caffeine*, http://smart.sns.it/caffeine, (Accessed June 15, 2018).

34. A. Salvadori, G. Del Frate, M. Pagliai, G. Mancini, and V. Barone, Int. J. Quantum Chem. **116**, 1731 (2016).

35. J. Jover and N. Fey, Chem. Asian J. **9**, 1714 (2014).

36. M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, et al., *Gaussian 16 Revision A.03* (2016), gaussian Inc. Wallingford CT.

37. C. Lee, W. Yang, and R. G. Parr, Phys. Rev. B. **37**, 785 (1988).

38. A. D. Becke, J. Chem. Phys. **98**, 5648 (1993).

39. P. J. Hay and W. R. Wadt, J. Chem. Phys. **82**, 270 (1985).

40. P. C. Hariharan and J. A. Pople, Theor. Chim. Acta **28**, 213 (1973).

41. M. M. Francl, W. J. Pietro, W. J. Hehre, J. S. Binkley, M. S. Gordon, D. J. DeFrees, and J. A. Pople, J. Chem. Phys. **77**, 3654 (1982).

42. T. Clark, J. Chandrasekhar, G. W. Spitznagel, and P. V. R. Schleyer, J. Comput. Chem. **4**, 294 (1983).

43. V. Barone, M. Cossi, and J. Tomasi, J. Chem. Phys. **107**, 3210 (1997).

44. M. Cossi, G. Scalmani, N. Rega, and V. Barone, J. Chem. Phys. **117**, 43 (2002).

45. J. Tomasi, B. Mennucci, and R. Cammi, Chem. Rev. **105**, 2999 (2005).

46. G. Scalmani and M. J. Frisch, J. Chem. Phys. **132**, 114110 (2010).

47. V. A. Rassolov, J. A. Pople, M. A. Ratner, and T. L. Windus, J. Chem. Phys. **109**, 1223 (1998).

48. S. Grimme, S. Ehrlich, and L. Goerigk, J. Comput. Chem. **32**, 1456 (2011).

49. V. Barone, J. Chem. Phys. **122**, 014108 (2005).

50. J. Bloino and V. Barone, J. Chem. Phys. **136**, 124108 (2012).

51. V. Barone, M. Biczysko, J. Bloino, M. Borkowska-Panek, I. Carnimeo, and P. Panek, Int. J. Quantum Chem. **112**, 2185 (2012).

52. V. Barone, M. Biczysko, and J. Bloino, Phys. Chem. Chem. Phys. **16**, 1759 (2014).

53. S. Rampino, *The Waverley program package*, http://www.srampino.com/code.html#Waverley, (Accessed April 11, 2018).

54. R. H. Crabtree, *The Organometallic Chemistry of the Transition Metals, 6th Edition* (John Wiley & Sons, Inc., Hoboken, 2014).

55. M. Fusè, I. Rimoldi, E. Cesarotti, S. Rampino, and V. Barone, Phys. Chem. Chem. Phys. **19**, 9028 (2017).

56. M. Fusè, I. Rimoldi, G. Facchetti, S. Rampino, and V. Barone, Chem. Commun. **54**, 2397 (2018).

57. C. Ericson, *Real-Time Collision Detection* (CRC Press, Boca Raton, 2004).