Finding density functionals with ML

Kieron Burke and friends
UC Irvine Chemistry & Physics

http://dft.uci.edu

A. Review of DFT

The electronic structure problem

- Use atomic units
- Born-Oppenheimer approximation
- All non-relativistic (but added back in)
- Wavefunctions antisymmetric and normalized
- Only discuss ground-state electronic problem here, but many variations.

Hamiltonian for N electrons in the presence of external potential $v(\mathbf{r})$:

$$\hat{H} = \hat{T} + \hat{V}_{\rm ee} + \hat{V},$$

where the kinetic and elec-elec repulsion energies are

$$\hat{T} = -\frac{1}{2}\sum_{i=1}^{N} \nabla_i^2, \qquad \hat{V}_{\rm ee} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j\neq i}^{N} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|},$$

and difference between systems is N and the one-body potential

$$\hat{V} = \sum_{i=1}^{N} v(\mathbf{r}_i)$$

Often $v(\mathbf{r})$ is electron-nucleus attraction

$$v(\mathbf{r}) = -\sum_{\alpha} \frac{Z_\alpha}{|\mathbf{r} - \mathbf{R}_\alpha|}$$

where α runs over all nuclei, plus weak applied $\mathbf{E}$ and $\mathbf{B}$ fields.

$$\{\hat{T} + \hat{V}_{\rm ee} + \hat{V}\}\,\Psi = E\,\Psi, \qquad E = \min_{\Psi} \langle \Psi | \hat{T} + \hat{V}_{\rm ee} + \hat{V} | \Psi \rangle$$

DFT method

1964: HK theorem: There exists F[n]

Define *fictitious* non-interacting electrons satisfying:

$$\left\{-\frac{1}{2}\nabla^2 + v_{\rm S}(\mathbf{r})\right\}\phi_j(\mathbf{r}) = \epsilon_j\phi_j(\mathbf{r}), \qquad \sum_{j=1}^{N}|\phi_j(\mathbf{r})|^2 = n(\mathbf{r}).$$

where $v_{\rm S}(\mathbf{r})$ is *defined* to yield $n(\mathbf{r})$.

Define $T_{\rm S}$ as the kinetic energy of the KS electrons, U as their Hartree energy and

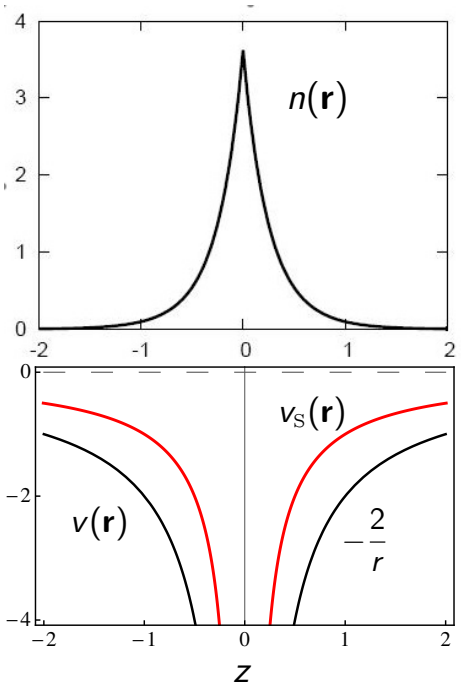
$$\mathsf{F}= T + V_{\rm ee} = T_{\rm S} + U + E_{\rm XC}$$

the remainder is the exchange-correlation energy.

Most important result of exact DFT:

$$v_{\rm S}(\mathbf{r}) = v(\mathbf{r}) + \int d^3r\frac{n(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|} + v_{\rm XC}[n](\mathbf{r}), \qquad v_{\rm XC}(\mathbf{r}) = \frac{\delta E_{\rm XC}}{\delta n(\mathbf{r})}$$

Knowing $E_{\rm XC}[n]$ gives closed set of self-consistent equations.



Orbital-free DFT: Approximate $T_s[n]$ directly, and go much, much faster.

Papers using DFT

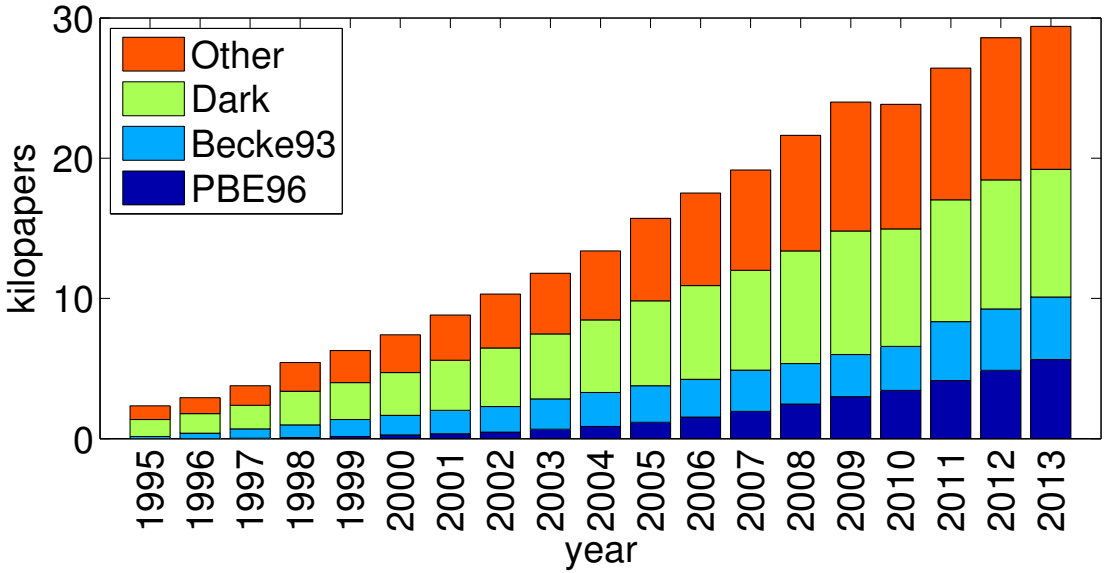


FIG. 2. The number of DFT citations has exploded (as have *ab initio* methods). PBE is the number of citations of of Ref. [22], and B88 of Ref. [18]. *Dark* indicates papers using either of these approximations without citing the original papers, while *other* is all other DFT papers. All numbers are estimates.

DFT: A Theory Full of Holes, Aurora Pribram-Jones, David A. Gross, Kieron Burke, Annual Review of Physical Chemistry (2014).

Semiclassical work in progress

- Almost exact exchange at almost no cost
  - for 1d boxes, do LDA calculation, then evaluate semiclassical exchange.
- Turning points
  - Finally derived and proved formulas in presence of turning points
- Asymptotic expansion for correlation
  - Gives new version of PBE, more accurate for atoms


Attila Cangi


Hardy Gross


Raphael Ribeiro

INSTITUTE FOR PURE AND APPLIED MATHEMATICS
Los Angeles, California
Semiclassical Origins of Density Functional Approximations
September 4 - 6, 2013

B. Strong correlation

- Ongoing project with Steve White at UCI
- Apply DMRG to continuum problems
- Understand limitations and failures of standard DFT approximations


Thomas E. Baker


Lucas Wagner


Miles Stoudenmire


Steve White

C. Machines

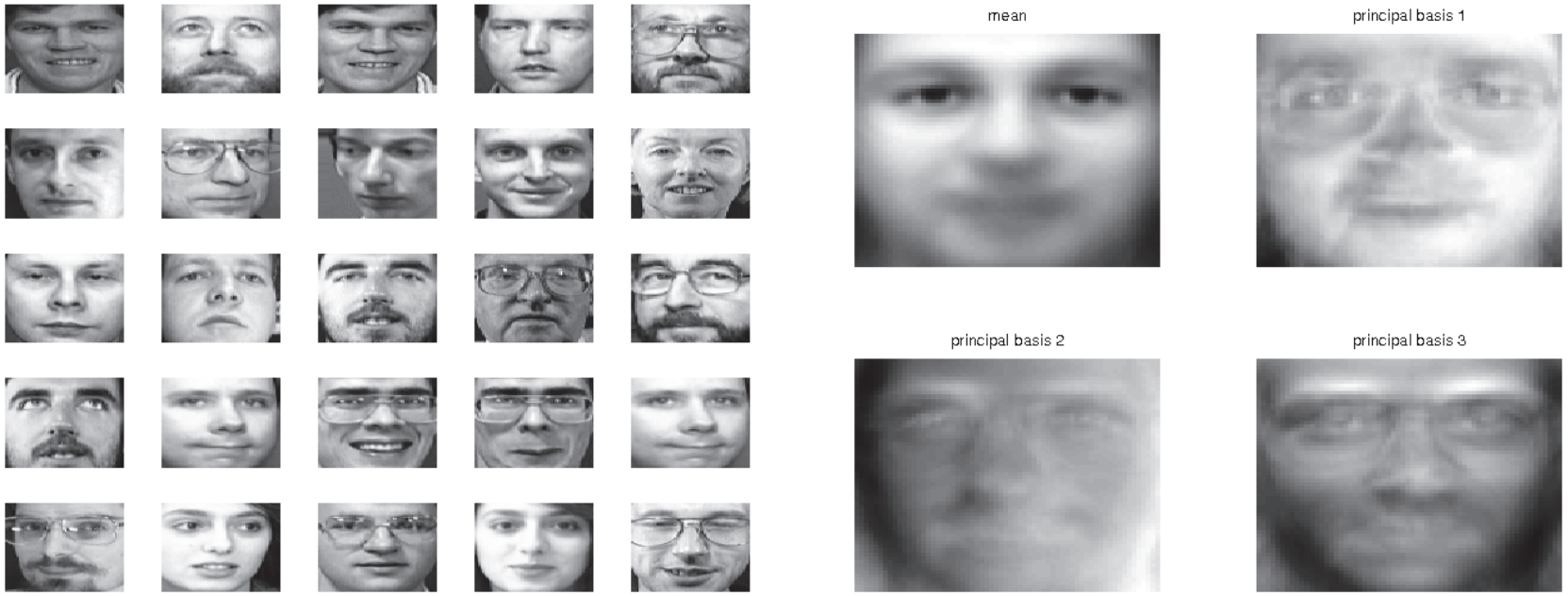


Figure 1.10 a) 25 randomly chosen 64×64 pixel images from the Olivetti face database. (b) The mean and the first three principal component basis vectors (eigenfaces). Figure generated by `pcaImageDemo`.

Machine learning

- Powerful branch of artificial intelligence
- Essentially fitting and interpolating
- Maps problem into much higher-dimension feature space, using a simple kernel
- Higher-dimension often means more linear
- Perform regression in feature space
- Project back to original problem

Kernel ridge regression

- Kernel ridge regression (KRR). Given $\{\mathbf{x}_j, f_j\}$

$$\hat{f}(\mathbf{x}) = \sum_{j=1}^{M} \alpha_j k(\mathbf{x}_j, \mathbf{x})$$

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2/(2\sigma^2))$$

length scale

- Minimize:

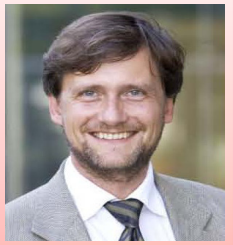
$$\mathcal{C}(\boldsymbol{\alpha}) = \sum_{j=1}^{M} (\hat{f}(\mathbf{x}_j) - f_j)^2 + \lambda^2 \|\alpha\|^2$$

noise level

$$\boldsymbol{\alpha} = (K + \lambda^2 I)^{-1}\mathbf{f}$$

ML applications in electronic structure

- Most with Klaus Mueller of TU Berlin, computer science.


Klaus-Robert Müller

- ML now being applied directly to, e.g., molecular energies from geometries for drug design, many by Matthias Rupp (U. Basel)


Matthias Rupp

- Our efforts are focused on finding $T_s[n]$ from examples, headed by John Snyder (Humboldt fellow at TU Berlin/MPI Halle)


John Snyder

Demo problem in DFT

- *N* non-interacting same-spin fermions confined to 1d box
- Define class of potential:

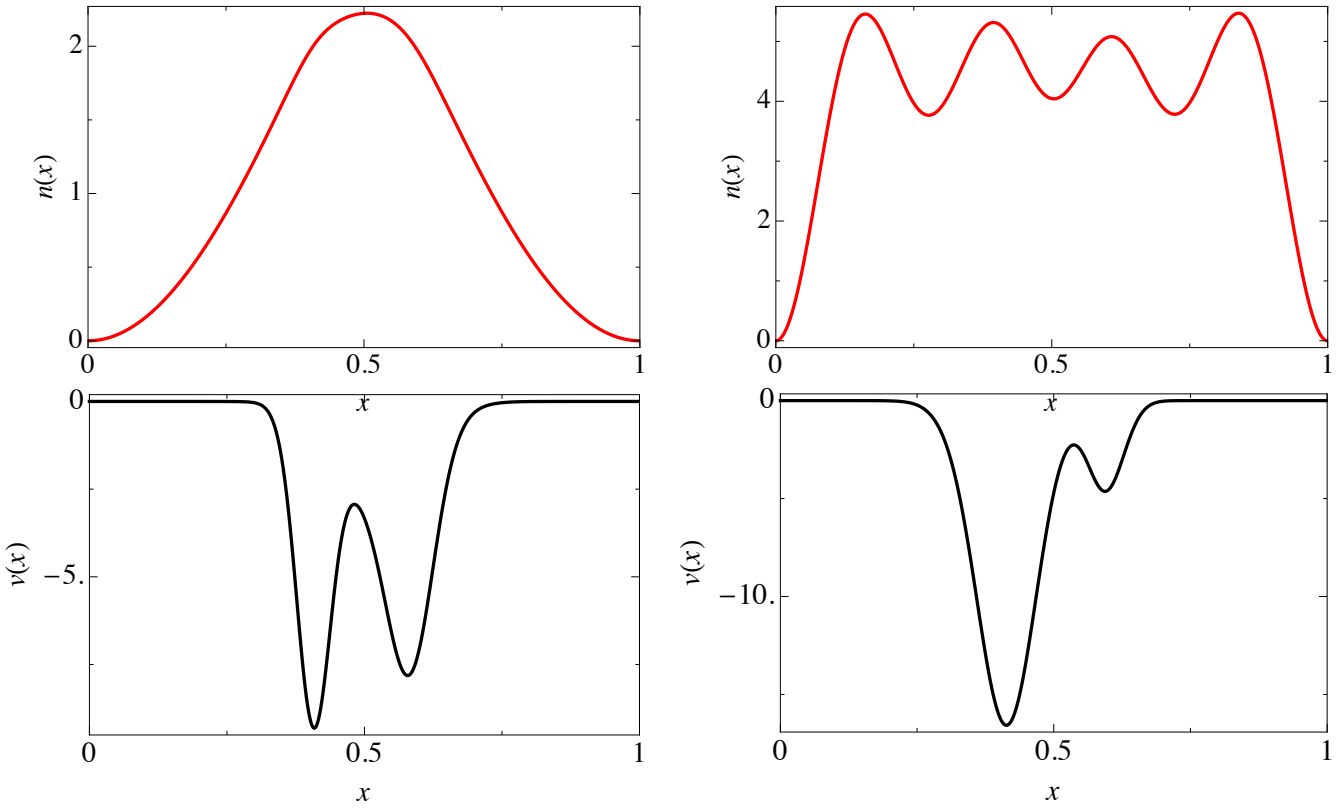
$$v(x) = -\sum_{i=1}^{3} a_i \exp(-(x-b_i)^2/(2c_i^2))$$

- Represent the density on a grid with spacing $\Delta x = 1/(G-1)$
- ML-DFA for KE:

$$\hat{T}(\mathbf{n}) = \bar{T} \sum_{j=1}^{M} \alpha_j k(\mathbf{n}_j, \mathbf{n})$$

Dataset

Generate 2000 potentials. Solve for up to 4 electrons.



Performance for T_s

kcal/mol

N	M	λ	σ	$\overline{\lvert\Delta T\rvert}$	$\lvert\Delta T\rvert^{\mathrm{std}}$	$\lvert\Delta T\rvert^{\mathrm{max}}$
1	40	2.4×10^{-5}	238	3.3	3.0	23.
	60	1.0×10^{-5}	95	1.2	1.2	10.
	80	6.7×10^{-6}	48	0.43	0.54	7.1
	100	3.4×10^{-7}	43	0.15	0.24	3.2
	150	2.5×10^{-7}	33	0.060	0.10	1.3
	200	1.7×10^{-7}	28	0.031	0.053	0.65
2	100	1.3×10^{-7}	52	0.13	0.20	1.8
3	100	2.0×10^{-7}	74	0.12	0.18	1.8
4	100	1.4×10^{-7}	73	0.078	0.14	2.3
1-4†	400	1.8×10^{-7}	47	0.12	0.20	3.6

LDA ~ 223 kcal/mol, Gradient correction ~ 159 kcal/mol

functional derivative?

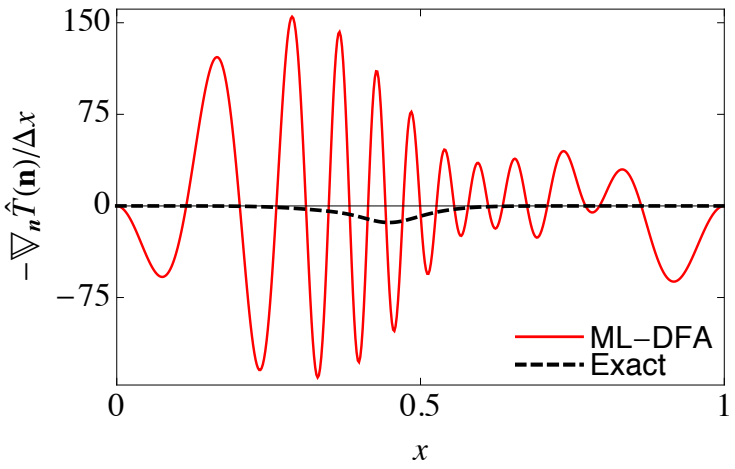
Exact

$$\frac{\delta T[n]}{\delta n(x)} = \mu - v(x)$$

ML-DFA

$$\frac{1}{\Delta x}\nabla_{\mathbf{n}}\hat{T}(\mathbf{n}) = \sum_{j=1}^{M} \alpha'_j(\mathbf{n}_j - \mathbf{n})k(\mathbf{n}_j, \mathbf{n})$$

$$\alpha'_j = \alpha_j/(\sigma^2\Delta x)$$



- Functionals are defined on infinite-dimensional spaces
- With finite interpolation, can always find bad directions
- Can we make a cruder definition that will work for our purposes?

Principal component analysis

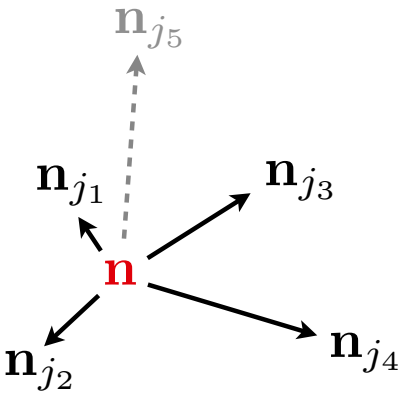
$$X = (\mathbf{n}_{j_1} - \mathbf{n}, \ldots, \mathbf{n}_{j_m} - \mathbf{n})^\top$$

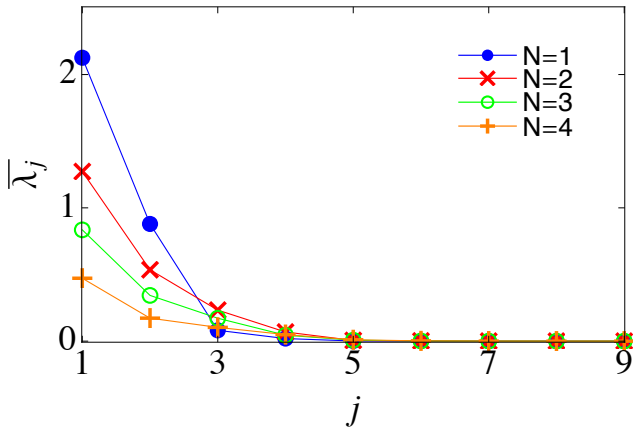
$$C = \frac{1}{m} X^\top X$$

$$\lambda_j, \mathbf{x}_j$$

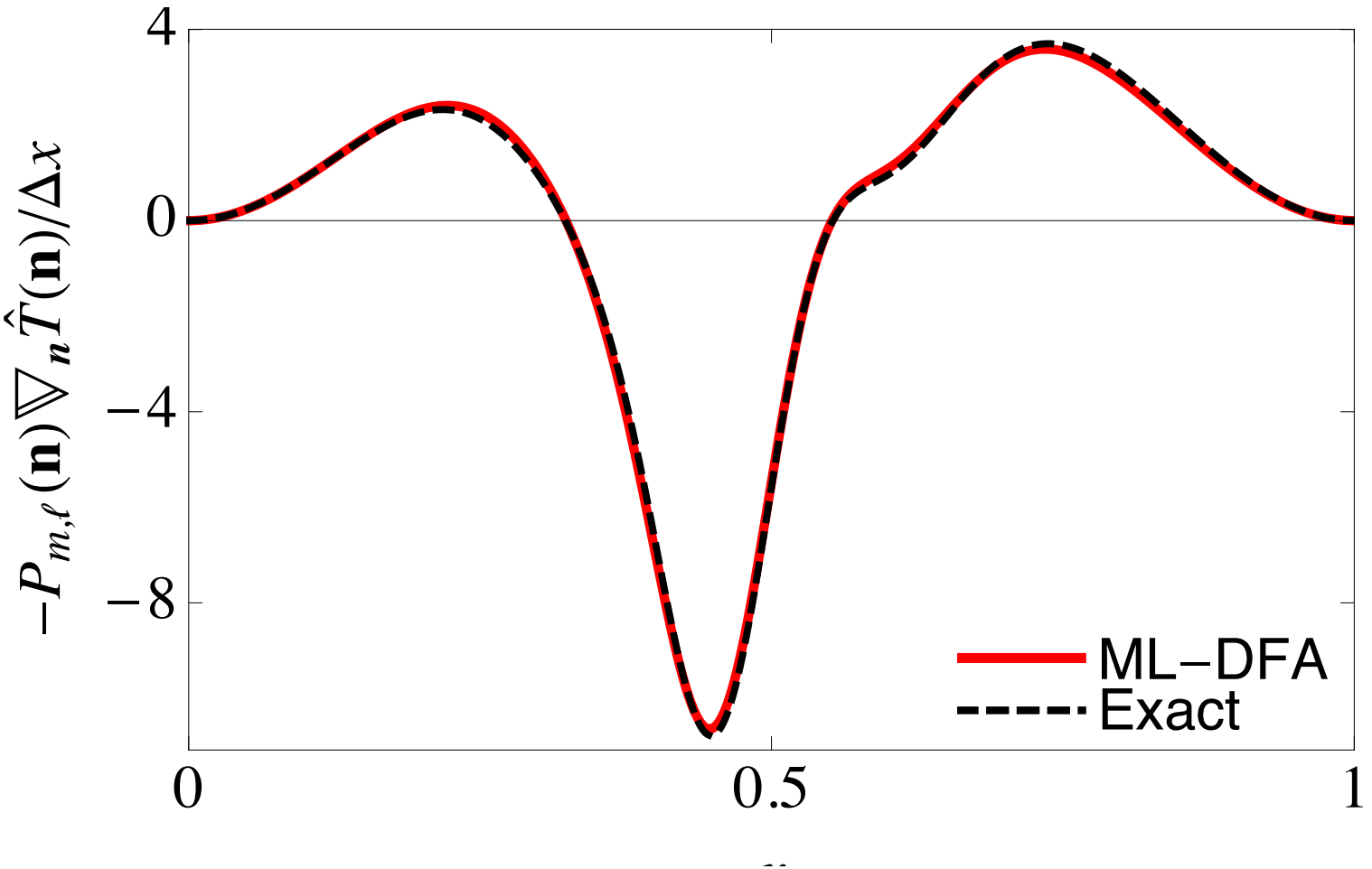
$$P_{m,\ell}(\mathbf{n}) = V^\top V$$

$$V = (\mathbf{x}_1, \ldots, \mathbf{x}_\ell)^\top$$





Projected functional derivative

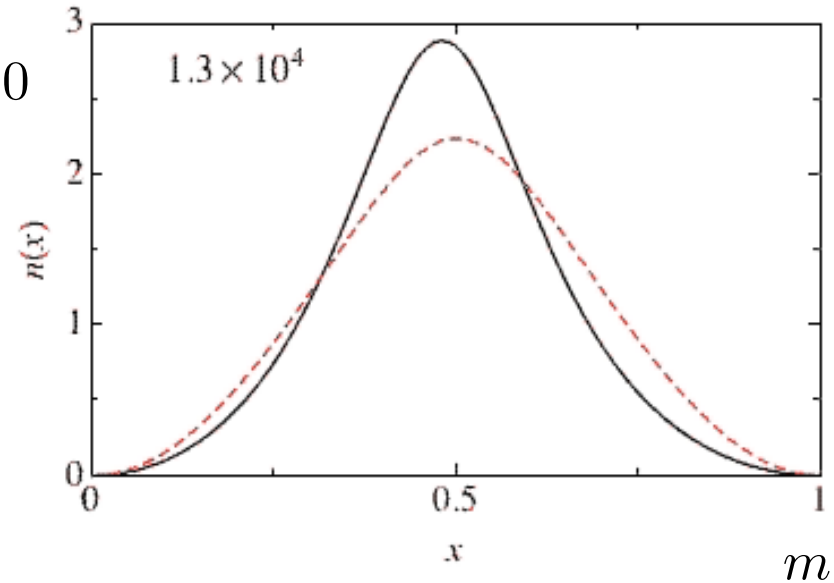


Constrained optimized density

- Gradient descent search:

$$\mathbf{n}^{(j+1)} = \mathbf{n}^{(j)} - \epsilon P_{m,\ell}(\mathbf{n}^{(j)})(\mathbf{v} + \nabla_{\mathbf{n}}\hat{T}(\mathbf{n}^{(j)})/\Delta x)$$

$N = 1$
$M = 100$

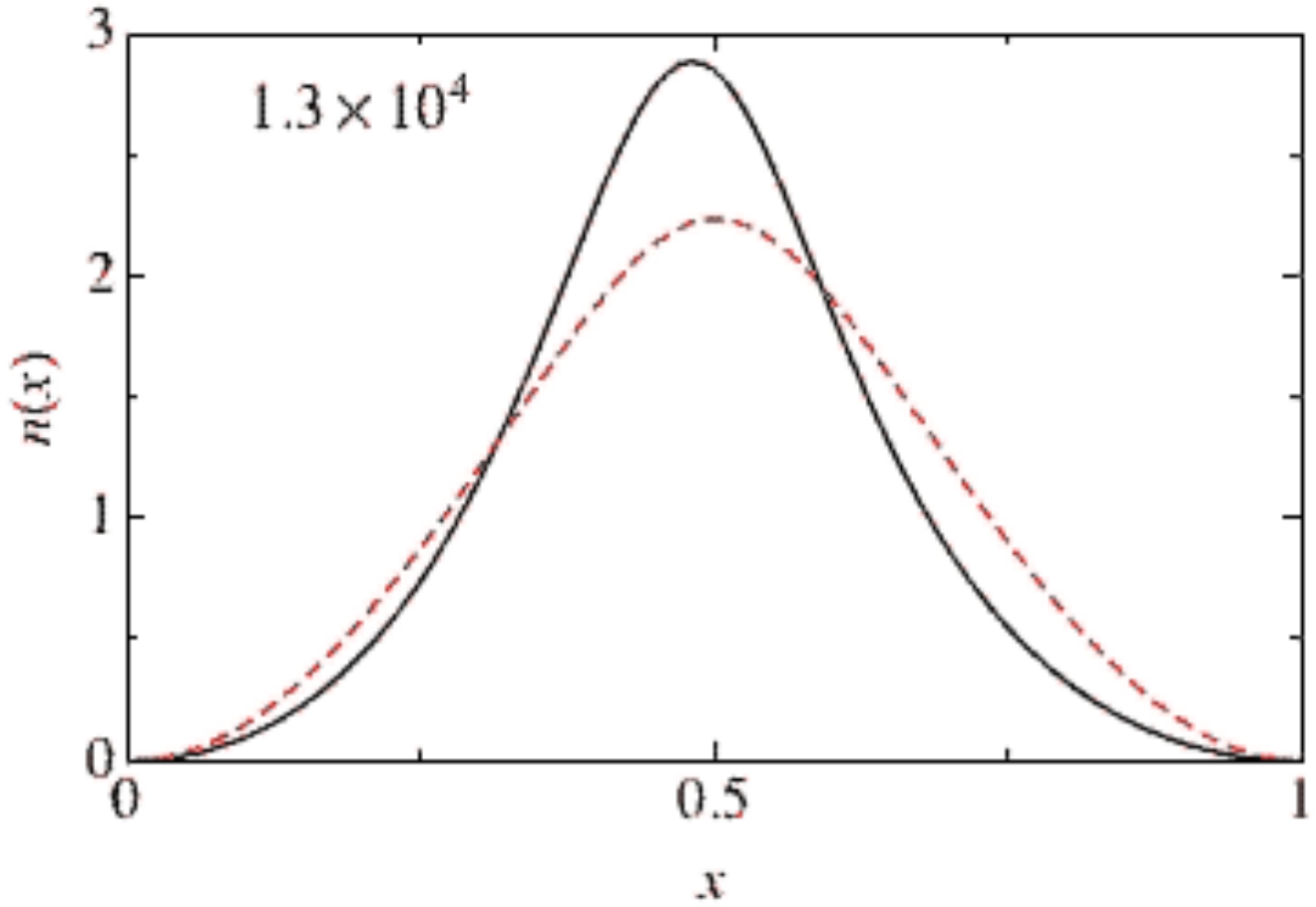


$\Delta T_j = 0.154$ kcal/mol
$\Delta T_j^{\mathrm{sc}} = 6.53$ kcal/mol

Ratio $= 43$

$m = 15, \ell = 5$

SC density movie



First ‘ML makes a functional’ paper

26	−22.3897314906413	3.751807395744921	0.0852624107335768	0.4453450848668749	7.559337237177321	0.05840909867970421	0.4990223467101435	4.729884445483556	0.0929218712306724	0.5590503374710945
27	5.595799173773367	4.522456026868127	0.03620201946841309	0.5050210060676815	1.885091277540955	0.06521780385848891	0.4961601035955842	9.47421945496329	0.0556740133396159	0.4834045749740091
28	1.94645582566549	2.751346713386582	0.0948195528516815	0.4901204212506473	7.863592227266858	0.0951797983951825	0.5196620049802691	9.51608680035275	0.07785377315463609	0.5806015592262945
29	−1.105559822533069	5.725277193011525	0.03097303116261406	0.5511177785567803	7.988513342309234	0.0928323179868588	0.5146415116367569	1.101098863639624	0.0882939028519791	0.5265108634130272
30	−3.886381532577754	4.95871172966177	0.06823254438777603	0.5645951313008806	9.08135225317634	0.04472856905840844	0.4299117124064201	8.24973637849809	0.08854431361856	0.5416193648861009
31	−1.341930696378109	6.451869235304859	0.05482804889270922	0.5694170176403242	4.416924333519995	0.04212208612720639	0.5897125000677342	8.86521100790565	0.0866581794842557	0.5311744230227635
32	−10.16467121571373	6.705508349541454	0.0934749698077619	0.4420693171728495	2.858371131410651	0.03722557506068529	0.4216580500297413	6.670274811535954	0.0977332470946321	0.538756085129185
33	−1.107633246744663	7.390220519784453	0.07970661674181083	0.5081940602224835	3.815123656821275	0.04965529863397879	0.497023655525965	9.69945245311718	0.07039341131023045	0.4990744291774679
34	10.39451871866851	1.478170300165134	0.03602109509210016	0.5799113381048154	2.105782960173057	0.0861928382367878	0.4812528298377607	7.010226383015251	0.03535327789989237	0.4969989664577149
35	−0.139925361351959	4.045190240200832	0.03579955552582782	0.5678362348598895	9.32386837735074	0.0858648976559044	0.5698752996855582	9.81895903799508	0.04704020956533079	0.5708692337579542
36	3.656124569605117	4.944179742822627	0.0931916525333049	0.4834640277977335	9.3847979419491	0.0840810063497973	0.4090254720704045	3.673880603434117	0.07158934023489803	0.5598196803643543
37	8.06734679112684	2.988178201634472	0.07482186947515447	0.4950144569660614	1.910770315838324	0.03877152903045122	0.5786659568499292	4.285706479098184	0.0594048755770145	0.4844513368813702
38	0.3011566621835722	8.78996207364355	0.04833452393773747	0.459203937360375	9.93467757777873	0.0973964303071365	0.4280349684453508	4.931948094146255	0.0969410287755626	0.45606603535297726
39	−0.2469178958324283	6.242957525364009	0.0989568314234112	0.4205662570527381	4.73983230004983	0.0450110225161678	0.559400017887981	9.11978947943194	0.06189826031162779	0.493100819449519
40	−5.704905642148452	4.447521595114223	0.06770523205457797	0.5047965897298675	3.987888630073714	0.03595136615254331	0.5929011044925536	5.22121823715344	0.0402943905467698	0.5141294116814418
41	0.1425795779687519	6.919606258393273	0.0821756368975312	0.5976799418614426	8.04099037018168	0.03129262110881438	0.4686530881925831	3.421695873368174	0.05166215779587051	0.4208560803818513
42	0.6843702894502789	8.27200587323167	0.0547013388431661	0.4453254786423237	7.433883810628654	0.04612202343826998	0.5269730370279639	8.3641619521824	0.07996879390804805	0.5036755718396144
43	−14.06571757369153	6.220712308892191	0.04765956545437692	0.4898959293071353	8.72249431241257	0.05059967705235588	0.4421274052875277	2.416276423317733	0.07373660930828208	0.5403850652535799
44	8.32346109443768	9.29716067662943	0.0656810603414827	0.4960066859011466	6.796263459296078	0.04037116107097755	0.4389885737869158	2.206183147865627	0.0804910989081056	0.5806221323132373
45	−5.252317079780442	3.424683176707511	0.0867529985575643	0.4322548560407479	6.859255296971943	0.0955962752949951	0.517620399931194	5.826144942819697	0.0910408442631135	0.4167014408128501
46	6.18641388898792	3.789578783769164	0.07617326304382499	0.5418490045761073	1.363177133649232	0.03197249245240319	0.5284999992906743	4.9879001955338	0.04126125322920861	0.4895167206754735
47	0.4821906326173532	1.195769022526768	0.04928749082970199	0.5286859472378347	8.76714063033632	0.0870852412295278	0.5989638152903285	2.489565114307663	0.05568685410478115	0.5787056957942209
48	5.707663438868804	4.058614230578957	0.04743875933294599	0.4981149890354436	1.015171237842736	0.085826684646538	0.5470349320774566	6.343764732282688	0.0878769436185823	0.5449452367497044
49	−0.1857958679022612	5.159763617905121	0.0850570422587659	0.5311491284465588	7.238990148015965	0.05867048711393652	0.5662410781222187	8.70600609334284	0.04088387548046445	0.5305143392409284
50	2.321594443893878	3.319181325761367	0.05391080867981007	0.5914435434600773	9.11776045377491	0.03583354001918901	0.4669413688494417	6.117669336479262	0.04553014156422459	0.4131324263527175
51	2.451069994722774	4.005247740100538	0.07620861067494449	0.5759225433332955	6.743529978394792	0.0967347280360008	0.4980631985926895	5.505849306951131	0.0971440363586883	0.45173467721102
52	−0.961350897030818	7.710615861039985	0.07131617125307138	0.4007362522722102	3.372061685311945	0.04579335436967417	0.4845503268496217	8.6021995929235	0.0634681851083368	0.4207242810824543
53	−10.67971490671366	4.728468083127391	0.06509200806253813	0.4959400801351127	9.81266481664175	0.0951346706530719	0.5047212015277842	3.180268423645401	0.06891913385037023	0.5758580789792047
54	1.375998039323297	2.740626958875014	0.06438323836566841	0.5077158168564866	1.813952930718781	0.0845523320869031	0.407247659218711	1.054506345314588	0.0998274789144316	0.5956344420441539
55	−4.330863793299111	3.209443235555092	0.05150806767094947	0.5932432195310449	4.682408324655706	0.03641376226612134	0.4491940043173043	9.57800470422746	0.089704223465467	0.4702887735348434
56	6.961655278126058	3.013659237176324	0.0802684211883349	0.5733460791404235	7.441047648005995	0.0924462539370359	0.5216671466381739	8.15403131934562	0.05611934350568269	0.4910483719005932
57	−5.30540083281513	2.088882156026381	0.07464091227469428	0.4740072756248339	4.228694369663863	0.03993178348974601	0.4402946238148492	4.912298433960689	0.06528359492264517	0.4923580906843156
58	3.395104400983763	7.973680986292	0.0911843583955236	0.5314007871406556	6.375772536276909	0.0933026364108379	0.447244215655568	7.387561983870292	0.0977643384454912	0.479824590293356
59	1.330224789015745	5.105339790874075	0.06331718284342962	0.5503804706506275	9.5740821641842	0.07463500644132846	0.5814481952546582	5.810342341443878	0.03327635127803191	0.5550216356360829
60	−0.2920604955263079	6.773414983805557	0.07595819905259191	0.5210834626114553	4.229974804881476	0.03532177545633547	0.5725615130215063	8.88452823254966	0.0932985710685446	0.5046388124857028
61	−26.41840990525562	6.41701161854283	0.06246944646172664	0.4258923441426071	1.379491105318868	0.06306367927818108	0.4531443030507202	2.189456974183496	0.098078005933584	0.5174384564079299
62	6.630393088825135	6.476325542442256	0.07599435647188182	0.4345528705798566	6.969363897694016	0.0666930203717610	0.517427894398595	3.961825639220393	0.0929115322244988	0.4621045370336795
63	3.065748817227897	1.300093759156537	0.06741885075599173	0.4866345116967513	6.942052331590217	0.07107963920756023	0.4930672781311588	7.834870626014938	0.06446115603870624	0.5367219218216266
64	−1.743105697039855	3.145607697855832	0.0415292						0.07636214080338054	0.4330291254331895
65	−1.631255302904965	3.121291975208031	0.0313356						0.0936192481138518	0.4523240987335903
66	−9.50982692642515	7.367027947839741	0.0739810						0.04986976063479617	0.5394709698394433
67	−4.496736421068983	8.56498650586069	0.0870950						0.05651923264880363	0.4832351547748815
68	−1.002977917543076	9.43121814362608	0.0368989						0.05983735955021298	0.48375600035001434
69	−9.23106545170694	8.4792969159459	0.0547214						0.06892772167363432	0.5908529311844791
70	14.99926304831282	4.708917711658332	0.0907364						0.05825558914206748	0.566434199642786
71	4.189181104505013	7.583635162152007	0.0936829						0.04763840334231468	0.5010633160850224
72	8.32837188138842	3.414315967033806	0.05054048464274665	0.5043307410524706	3.070996567281799	0.06968276328974509	0.5702669702356848	5.360783854112533	0.0388888383066967	0.5784155467566237
73	−21.84200888491582	1.625552430388586	0.04546236831654488	0.5426078576324179	4.091176923293506	0.03836885039652842	0.504879512949007	4.527748284314512	0.05975894102927981	0.4670598506496871
74	6.604738977945651	3.42187897931262	0.0894449950697187	0.4941485628974172	4.96749741331263	0.06031151123783582	0.4119028637456998	4.581927607849956	0.0871948963692279	0.4176805754532423
75	15.27988423246628	6.229538426957454	0.0821521091215863	0.4118748449252315	2.377028497509498	0.0963110933106458	0.4746736556808109	5.398545541594487	0.0822574656780643	0.5258947233190487

Finding Density Functionals with Machine Learning John C. Snyder, Matthias Rupp, Katja Hansen, Klaus-Robert Müller, Kieron Burke, Phys. Rev. Lett. **108**, 253002 (2012)

Li Li's poster

Understanding Machine-learned Density Functionals

Li Li* (李力), John C. Snyder, Isabelle M. Pelaschier, Jessica Huang, Uma-Naresh Niranjan, Paul Duncan, Matthias Rupp, Klaus-Robert Müller, Kieron Burke

*li.li@uci.edu | Department of Physics and Astronomy, University of California, Irvine, CA 92697

Introduction

Burke K. JCP, 2012, 136(15): 150901.

Is there a data-driven method that can be applied to approximate density functionals?

Density functional theory (DFT) is the most popular electronic structure calculation method in the world. In principle, the mapping of the electron density to energy is exact, but in practice, both the kinetic energy and the energy of the interaction between electrons must be approximated.

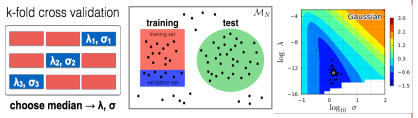
In the past few decades, development of both empirical and non-empirical functionals requires great intuition built on years of experience, as well as painstaking trial and error. Recently, some of us have approached this problem by a completely new data-driven method called machine learning approximation (MLA). Comparing to traditional approximation, MLA has (i) lower computational cost and (ii) significantly higher accuracy.

Machine Learning

Kernel Ridge Regression is a powerful kernel-based ML method to find non-linear pattern in high dimensional space.

- ***training set*** $\{n_j, T[n_j]\},\ j=1,2,...,N_T \Rightarrow T^{ML}[n] = \sum_{j=1}^{N_T} \alpha_j k[n, n_j]$
- ***kernel trick***
  we define a mapping for input space to feature space: $\Phi : \chi \to F$
  kernel is defined as a inner product: $k(x,y) = \langle \Phi(x), \Phi(y) \rangle$
  a low dimensional nonlinear fitting is tranfered to a high dimensional linear fitting problem.
  Gaussian kernel: $k[n_i, n_j] = \exp\left(-\|n_i - n_j\|^2 / 2\sigma^2\right)$
- ***cost function***
  $\mathcal{C}(\alpha) = \sum_{j=1}^{N_T} (T^{ML}[n_j] - T[n_j])^2 + \lambda \alpha^T \mathbf{K} \alpha$
  $\Rightarrow \alpha = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{T}$
- ***Hyperparameters***
  ✧ noise level λ
  ✧ length scale σ

Optimization



The ML model is built from the training set and the hyperparameters are optimized by minimizing the error on the validation set. The test set is never touched until the weights and hyperparameters have been determined. Then and only then, the generalization ability of the model can be assessed with the test data.

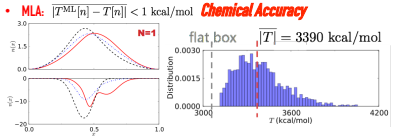
Model System

Consider N non-interacting same-spin fermions subject to a smooth external potential in one dimension,

$$v(x) = \sum_{j=1}^{3} a_j \exp(-(x-b_j)^2/(2c_j^2)) \quad \begin{matrix} 1 < a_j < 10 \\ 0.4 < b_j < 0.6 \\ 0.03 < c_j < 0.1 \end{matrix}$$

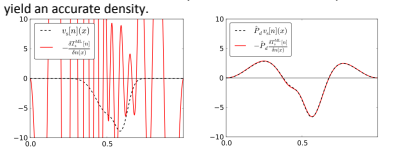
with hard wall on both side. We are going to approximate the kinetic energy function T[n].

- Local density approximation (LDA): $|T^{loc}[n] - T[n]| = 217$ kcal/mol
- MLA: $|T^{ML}[n] - T[n]| < 1$ kcal/mol ***Chemical Accuracy***



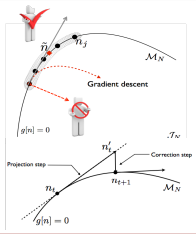
Challenge of Finding Density

Thus far, we have focused on the discussion of the performance of the MLA evaluated on exact densities. However, in order for a functional to be useful, it must also predict the ground-state density. An accurate functional derivative is necessary in order to solve Euler equation and yield an accurate density.



raw functional derivative | projected functional derivative

Constraint Manifold Searching



- Evaluate the functional derivative
  $\frac{\delta E^{ML}[n]}{\delta n(x)} = \frac{\delta T^{ML}[n]}{\delta n(x)} + v(x)$ at $n = n_t$.
- Compute projection operator by Principal Component Analysis (PCA)
  $P[n] = \sum_{j=1}^{d} \mathbf{u}_j \mathbf{u}_j^T$
- Gradient descent on tangent space
  $n_t'(x) = n_t(x) - \epsilon \tilde{P}[n_t] \frac{\delta E^{ML}[n]}{\delta n(x)}\Big|_{n=n_t}$
- correction
  $n_{t+1}(x) = n_t'(x) - (1 - \tilde{P}[n_t'])(n_t' - \bar{n}[n_t'])$

Results

N	N_T	λ x10^14	σ	$\|T^{ML}[n]-T[n]\|$ mean	$\|T^{ML}[n]-T[n]\|$ max	$\|T^{ML}[\bar{n}]-T[n]\|$ mean	$\|T^{ML}[\bar{n}]-T[n]\|$ max
1	40	50	4.2	1.9	30	15	120
1	60	10	1.8	0.62	11	3.0	19
1	80	54	1.5	0.23	3.1	1.1	11
1	100	4.5	1.6	0.13	3.5	1.4	16
2	100	1.0	2.2	0.14	1.7	1.25	5.0
3	100	1.9	2.5	0.13	1.7	1.11	8.3
4	100	1.4	2.7	0.08	2.6	1.12	9.8
1-4	400	1.7	2.2	0.12	3.0	1.28	12.6

*energy unit: kcal/mol

Conclusion

We used a simple model as a proof of principle, to investigate how standard methods from ML can be applied to DFT.

This novel method works to produce highly accurate approximate functionals. The performance of MLA increases with larger training set.

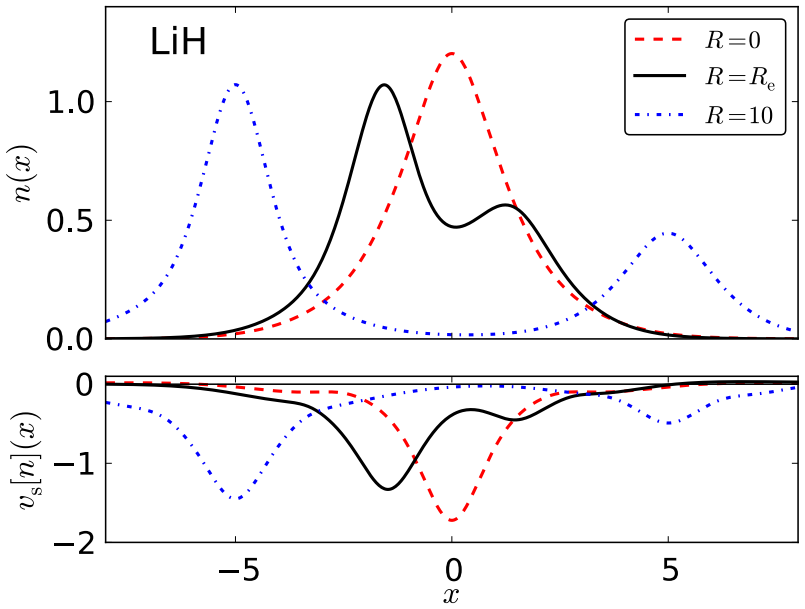
With a modified orbital-free DFT, we can also obtain highly accurate self-consistent densities and energies.

The authors thank for the support from NSF Grant No. CHE-1240252 (JS, KB). KRM thanks the BK21 Plus Program by NRF Korea, DFG and the Einstein Foundation. Correspondence to Li Li and K.-R. Müller.

[1] Understanding Machine-learned Density Functionals, *Phys. Rev. A (submitted)*
[2] Orbital-free Bond Breaking via Machine Learning, *J. Chem. Phys.* **139**, 224104 (2013)
[3] Finding Density Functionals with Machine Learning, *Phys. Rev. Lett.* **108**, 253002 (2012)

Bond-breaking with ML

- Performed many 1d KS calculations of diatomics as function of bond length, using LDA with soft-Coulomb repulsion, including several with more than 2 electrons



Orbital-free Bond Breaking via Machine Learning John C. Snyder, Matthias Rupp, Katja Hansen, Leo Blooston, Klaus-Robert Müller, Kieron Burke, J. Chem. Phys. **139**, 224104 (2013)

Constrained optimal density

- Convergence of constrained optimal density with # of training points.

Kernels, Pre-Images and Optimization John Snyder, Sebastian Mika, Kieron Burke, Klaus-Robert Müller, Chapter in Empirical Inference - Festschrift in Honor of Vladimir N. Vapnik (2013)

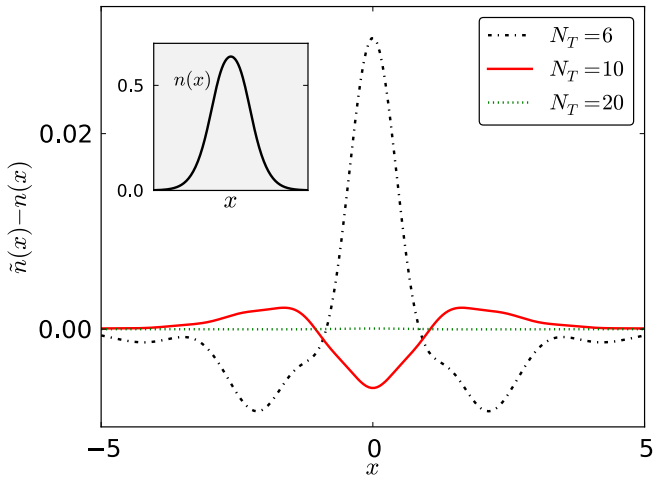


FIG. 7. Difference between the constrained optimal density $\tilde{n}(x)$ and the KS density $n(x)$ for various numbers of training densities N_T. The error decreases uniformly for all x. The system is H_2 at equilibrium bond length. The inset shows the KS density.

Types of errors in DFT

- $\Delta E_F = \bar{E}_{xc}[n]-E_{xc}[n]$
- $\Delta E_D = \bar{E}_{xc}[\tilde{n}]-\bar{E}_{xc}[n]$
- $\Delta E = \Delta E_F + \Delta E_D$
- Error analysis of energies in kcal/mol as a function of R with different numbers of training data, on constrained optimal densities

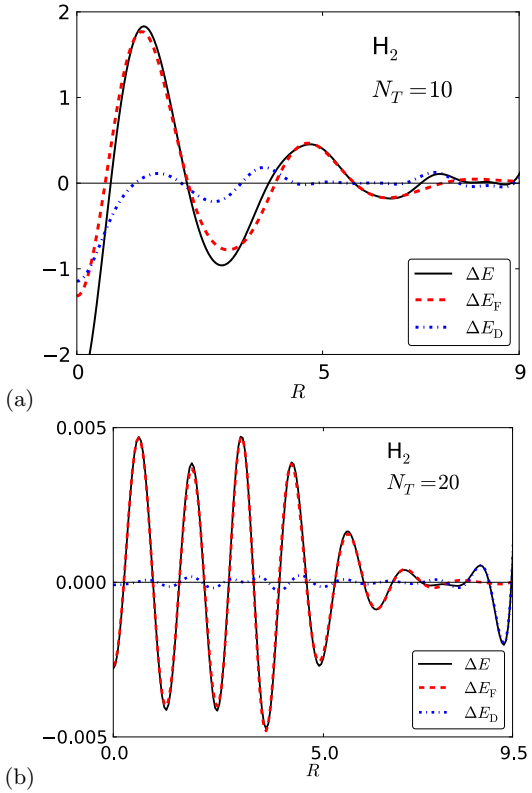


FIG. 8. The total error of the model and the functional- and density-driven errors ΔE_F and ΔE_D for H_2 with (a) 10 and (b) 20 training densities.

Functional derivatives and densities

- How can we get accurate densities from lousy derivatives?
- Once solution density is within interpolation manifold, simply constrain derivative to stay on that manifold
- Analogy:
  - Problem: find global minimum of 2D surface, given exact data along a 1D curve in that surface that passes through the minimum.
  - Solution: Make sure you stay on the path.
- PS: Inspired density-corrected DFT, which corrects many self-interaction errors!

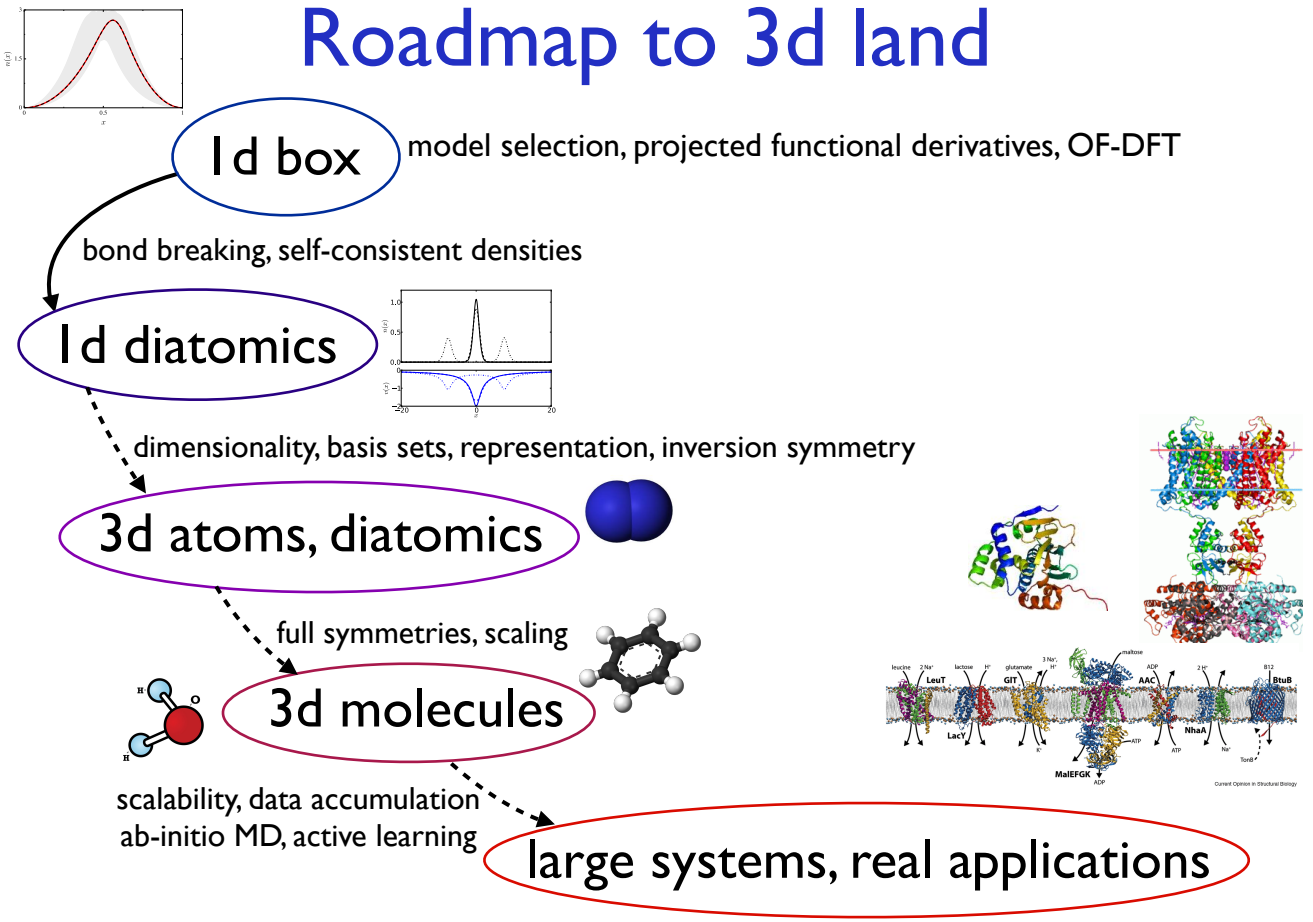
Understanding and reducing errors in density functional calculations Min-Cheol Kim, Eunji Sim, Kieron Burke, Phys. Rev. Lett. **111**, 073003 (2013).

Ions in solution: Density corrected density functional theory (DC-DFT) Min-Cheol Kim, Eunji Sim, Kieron Burke, The Journal of Chemical Physics **140**, 18A528 (2014)

Conceptual relationship

- ML works when
  a) There's a rule
  b) Rule is too complicated for humans
  c) There's data
- HK theorems say
  a) There is a functional
  b) It cannot be given explicitly, exactly
  c) Examples give exact values
- More important, practically:
  – In chemistry and materials, we only care about solutions to an absurdly small fraction of possible problems, i.e., Coulomb potentials at various positions, so underlying dimensionality of solutions is very small, just solving differential equation is hard.

Road map back to reality



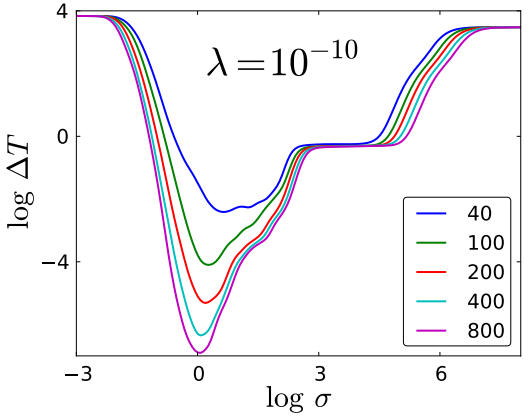
D. Simplified example


Kevin Vu

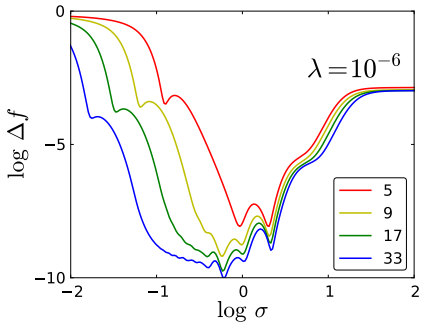

Li Li

Kevin's paper: from functions to functionals

- Plot error as a function of hyperparameters



- Repeat for fitting f(x)= cos x



- Curves have roughly the same "valley" shape for all N_T
- Bottom of the valley is an order of magnitude deeper than the walls
- These valleys are nearly identical in shape for sufficiently large N_T, which indicates that this particular feature arises in a systematic manner as N_T increases

D. Cross-fertilization

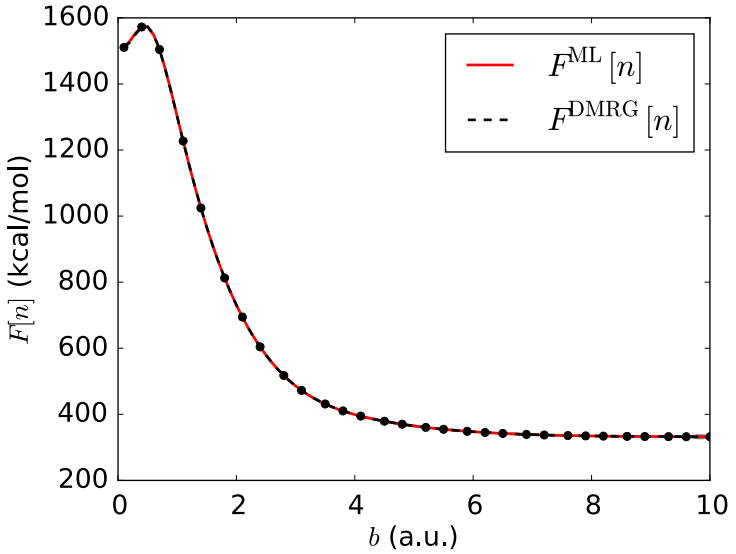
- All preliminary results

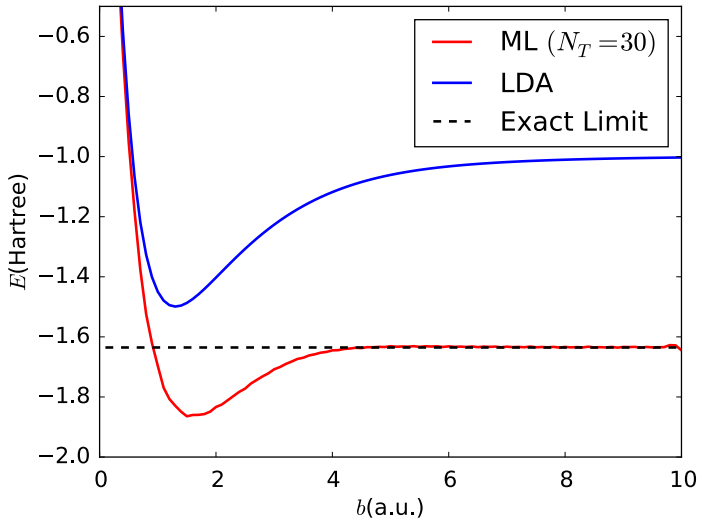

Li Li


Thomas E. Baker

DMRG meets DFT meets ML

- Ran H_4 with fixed separations *b* using DMRG
- Use 30 values of *b* to train ML version of exact F[n]
- Yields accurate exact binding energy curve self-consistently.





Summary

- ML of functionals works in model cases to produce highly accurate approximate functionals
- Totally different approach from anything before
- ML can even
  - find accurate densities
  - say when it will work within tolerance (makes Klaus nervous)
  - break bonds
  - Do the full functional
- But
  - only demonstrated in 1d
  - Need to do arbitrary-sized system (representation question)
- Thanks to
  - Students: Li Li, John Snyder, Kevin Vu, Isabelle Pelaschier
  - Collaborators: Klaus Mueller, Matthias Rupp, Katia Hansen
  - Funders: NSF from chem, DMR, math