Implementation and measurement of new Poisson solvers in `Octopus`

Lyon
October 22-23

Joseba Alberdi-Rodriguez
University of the Basque Country UPV/EHU

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

ALDAPA
nano-bio spectroscopy group

Outline

From where we came?

Where we go?

Survey

Conclusions

Outline

From where we came?

Where we go?

Survey

Conclusions

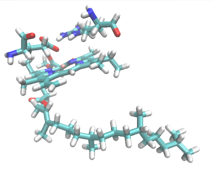
Introduction

This presentation is based on a paper that we are going to publish "*A survey of the parallel performance and accuracy of Poisson solvers for electronic structure calculations*" of the following authors:

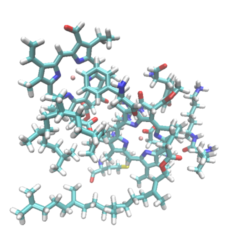
- Pablo García Risueño
- Joseba Alberdi-Rodriguez
- Micael J. T. Oliveira
- Xavier Andrade
- Michael Pippig
- Javier Muguerza
- Agustin Arruabarrena
- Angel Rubio

What do we want?

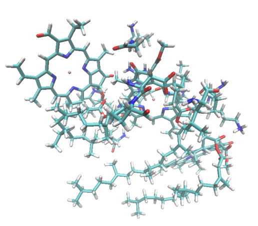
- Simulate efficiently chlorophyll molecule



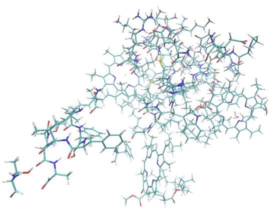
180 atoms



441 atoms



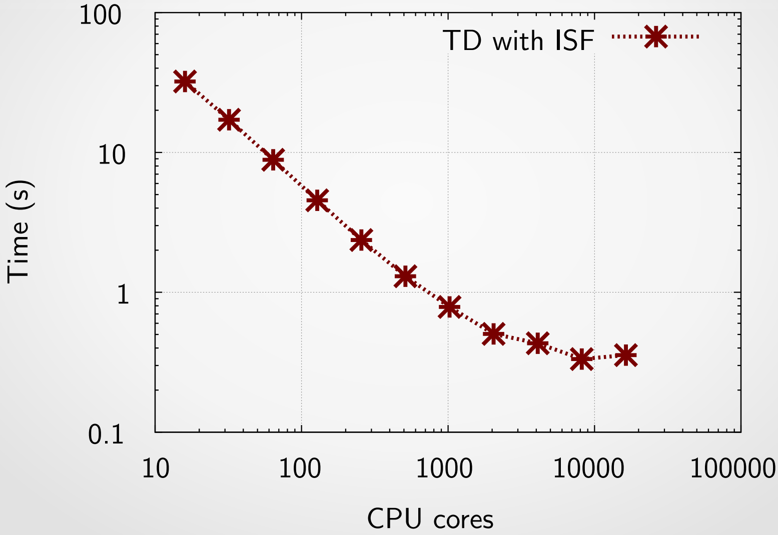
650 atoms



1365 atoms

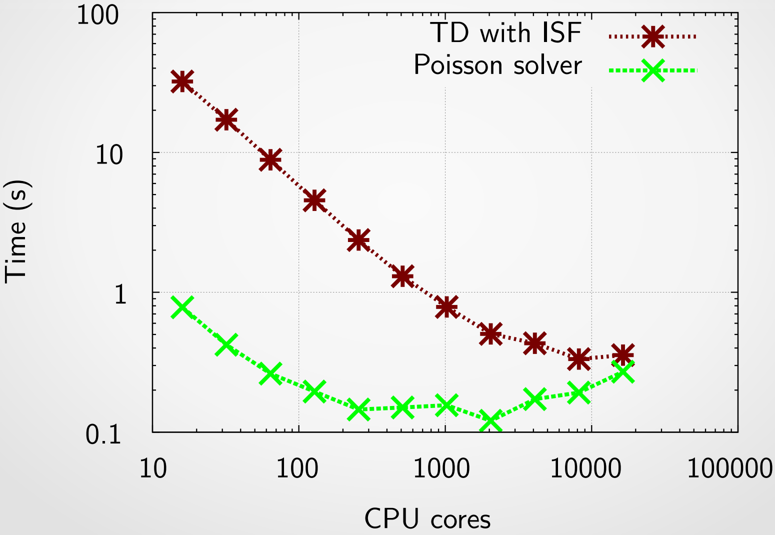
What was the problem?

- Time-dependent runs did not scale ideally [?]

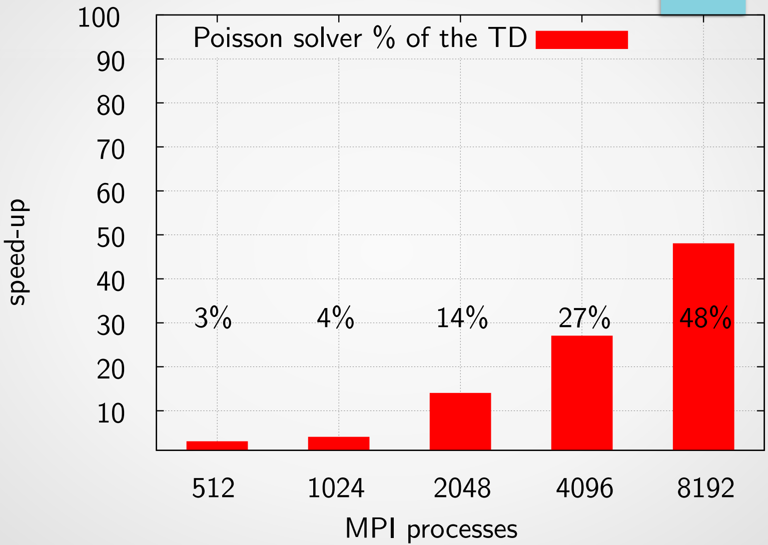


Poisson solver did not scale,

- Blue Gene/P



or; Not optimised Poisson solver



Outline

Possible solution

Change to a new scalable Poisson solver

Possible solution

Change to a new scalable Poisson solver

- Fast multipole method

Possible solution

Change to a new scalable Poisson solver

- Fast multipole method
- Parallel fast Fourier transform

Possible solution

Change to a new scalable Poisson solver

- Fast multipole method
- Parallel fast Fourier transform

So, we have implemented two new Poisson solvers based on parallel FFT [**?**] fast multipole method (FMM) [**?**] libraries.

Parallel fast Fourier transform

Based on serial FFT

- Works with parallelepiped grid shapes only
- Padding from OCTOPUS grid needed
  - Done with `MPI_Gather` and `MPI_Scatter`
  - Never better than constant time.

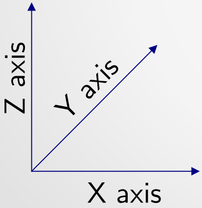
PFFT implementation

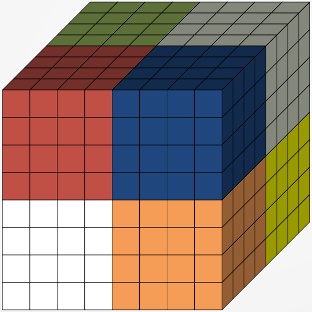
- Also, parallelepiped grid shape
- But, with a new grid partitioning (next slide)
- Implemented highly parallel data-movement, using `MPI_Alltoall`

PFFT grid partitioning

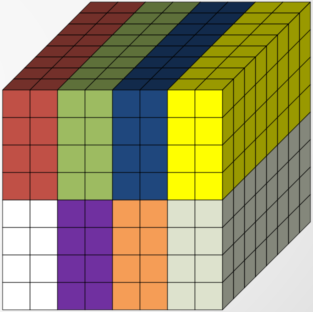
Simplified domain decomposition of the simulation meshes.

- A) OCTOPUS mesh with a 3D domain decomposition
- B) PFFT mesh with a 2D decomposition.



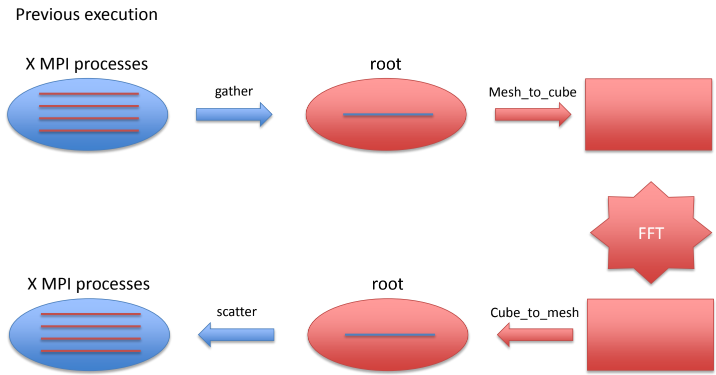


A) Octopus mesh

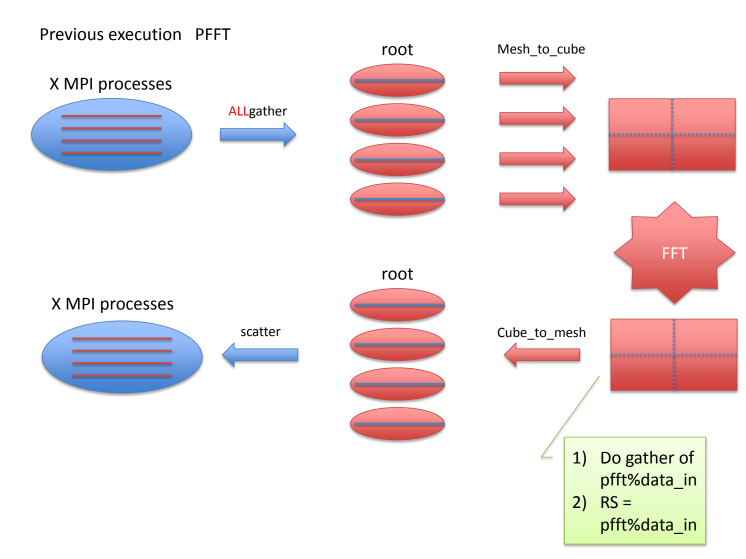


B) PFFT mesh

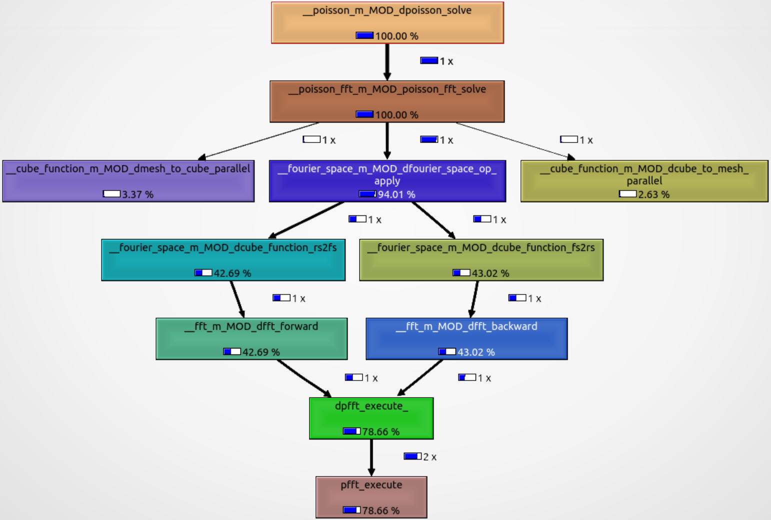
Serial FFT solver



Parallel FFT solver



Parallel FFT solver execution



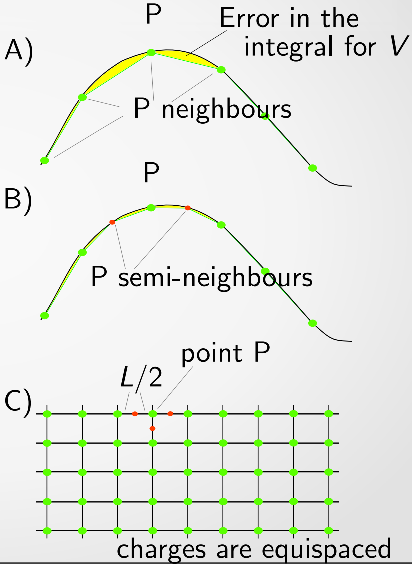
Fast multipole method

- Usage of an external library, developed in JSC
- Use of the adaptive grid shape of OCTOPUS
- Implementation of a correction term
- $\mathcal{O}(N)$ complexity

Fast multipole method (II)

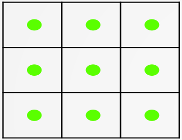
Scheme of how the inclusion of semi-neighbours of point P

- A) Scheme without semi-neighbours.
- B) considering semi-neighbours of point P

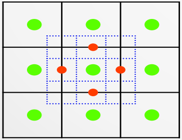


Fast multipole method (III)

2D example of the position of cells containing semi neighbours.$\vec{r}_0$-centred cell itself.

A) 

Original box: all cell's size is L^2 (2D)

B) 

New: Semi-neighbours cell sizes are $L^2/4$
Side cells sizes are $7/8L^2$
Central cell size is $L^2/2$
(all in 2D)

Outline

Survey

We have measured the accuracy and the performance of the 6 available Poisson solver of OCTOPUS. Those are compared solvers:

- PFFT
- ISF
- FMM
- CG corrected
- Multigrid

Accuracy

We measured the accuracy of a Poisson solver, E_{pot} and E_{energy}, as follows:

$$E_{pot} := \frac{\sum_{ijk} |v_a(\vec{r}_{ijk}) - v_n(\vec{r}_{ijk})|}{\sum_{ijk} |v_a(\vec{r}_{ijk})|} ,$$

$$E_a = \frac{1}{2} \sum_{ijk} \rho(\vec{r}_{ijk}) v_a(\vec{r}_{ijk}) ,$$

$$E_n = \frac{1}{2} \sum_{ijk} \rho(\vec{r}_{ijk}) v_n(\vec{r}_{ijk}) ,$$

$$E_{energy} := \frac{E_a - E_n}{E_a} ,$$

where;

- $\vec{r}_{ijk}$: all the points of the analysed grid
- v_a: analytically calculated potential
- v_n: calculated potential calculated
- E_a: analytical Hartree energy
- E_n: numerical Hartree energy

Accuracy (E_{pot} error)

E_{pot} errors of different Poisson solvers in the calculation of the Hartree potential created by a Gaussian charge distribution represented on a grid of edge $L_e = 15.8$ Å and spacing 0.2 Å.

L_e (Å)	PFFT	ISF	FMM	CG	Multigrid
7	$9{\cdot}10^{-3}$	$4{\cdot}10^{-3}$	$4{\cdot}10^{-3}$	$4{\cdot}10^{-3}$	$4{\cdot}10^{-3}$
10	$1{\cdot}10^{-4}$	$2{\cdot}10^{-5}$	$8{\cdot}10^{-5}$	$3{\cdot}10^{-5}$	$2{\cdot}10^{-5}$
15.8	$6{\cdot}10^{-5}$	$2{\cdot}10^{-7}$	$1{\cdot}10^{-4}$	$5{\cdot}10^{-5}$	$6{\cdot}10^{-6}$
22.1	$1{\cdot}10^{-5}$	$4{\cdot}10^{-10}$	$3{\cdot}10^{-4}$	$3{\cdot}10^{-4}$	$2{\cdot}10^{-6}$
25.9	$4{\cdot}10^{-7}$	$7{\cdot}10^{-10}$	$3{\cdot}10^{-4}$	$5{\cdot}10^{-3}$	$4{\cdot}10^{-7}$
31.7	$4{\cdot}10^{-8}$	$1\ {\cdot}10^{-9}$	$2{\cdot}10^{-4}$	$1{\cdot}10^{-2}$	$4{\cdot}10^{-7}$

Accuracy (E_{energy} error)

E_{energy} errors of different Poisson solvers in the calculation of the Hartree potential created by a Gaussian charge distribution represented on a grid of edge $L_e = 15.8$ Å and spacing 0.2 Å.

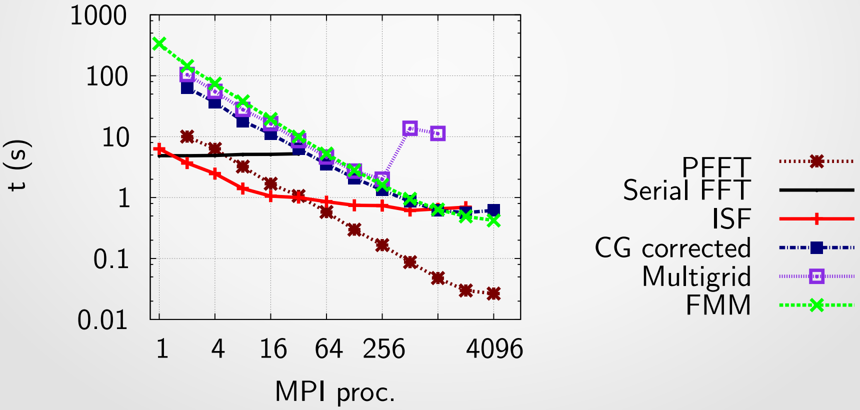
L_e (Å)	PFFT	ISF	FMM	CG	Multigrid
7	$2{\cdot}10^{-3}$	$2{\cdot}10^{-3}$	$2{\cdot}10^{-3}$	$2{\cdot}10^{-3}$	$2{\cdot}10^{-3}$
10	$9{\cdot}10^{-6}$	$9{\cdot}10^{-6}$	$4{\cdot}10^{-6}$	$4{\cdot}10^{-6}$	$9{\cdot}10^{-6}$
15.8	$2{\cdot}10^{-12}$	$3{\cdot}10^{-7}$	$3{\cdot}10^{-6}$	$3{\cdot}10^{-5}$	$5{\cdot}10^{-6}$
22.1	$<2{\cdot}10^{-12}$	$1{\cdot}10^{-11}$	$3{\cdot}10^{-6}$	$4{\cdot}10^{-4}$	$-1{\cdot}10^{-6}$
25.9	$<2{\cdot}10^{-12}$	$9{\cdot}10^{-12}$	$-4{\cdot}10^{-6}$	$5{\cdot}10^{-3}$	$2{\cdot}10^{-7}$
31.7	$<4{\cdot}10^{-13}$	$8{\cdot}10^{-12}$	$-3{\cdot}10^{-6}$	$1{\cdot}10^{-2}$	$-5{\cdot}10^{-7}$

Performance

- We have used 2 different architectures
  - x86-64
    - Curie
    - Corvo
  - Blue Gene/P
    - Jugene
    - Genius
- We have run hartree test and measured the time of the Poisson solver

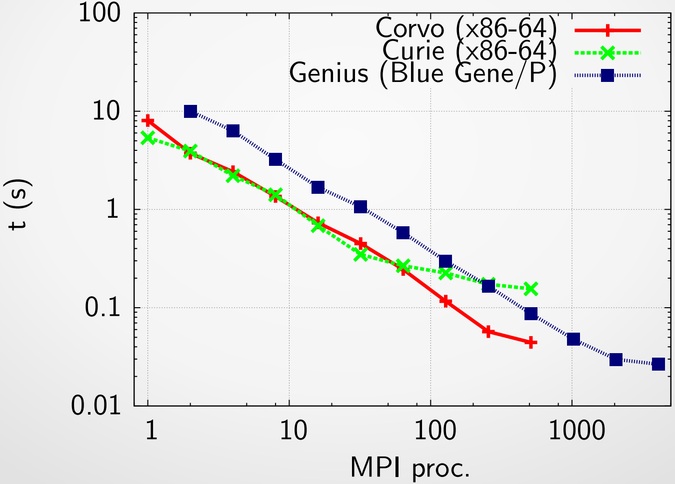
Poisson solver, Blue Gene/P

Execution-times for the calculation of the Hartree potential created by a Gaussian charge distribution on a Blue Gene/P machine as a function of six different Poisson solvers and of the involved number of MPI processes.

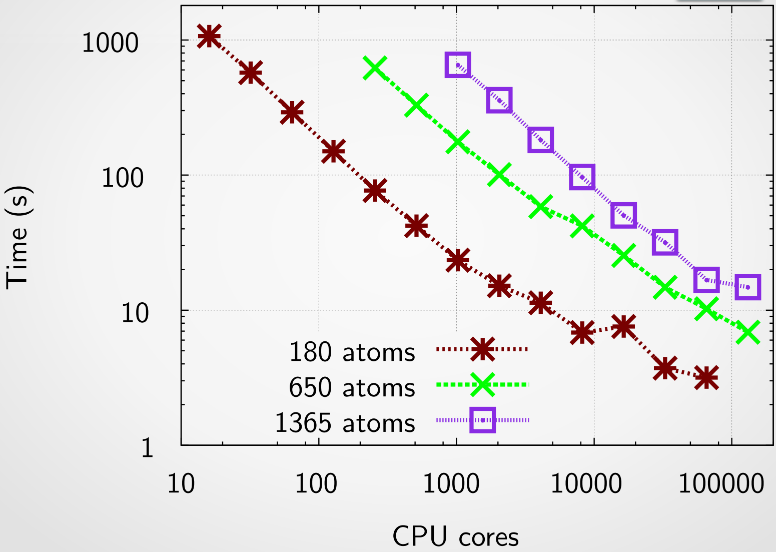


Poisson solver, machine comparison

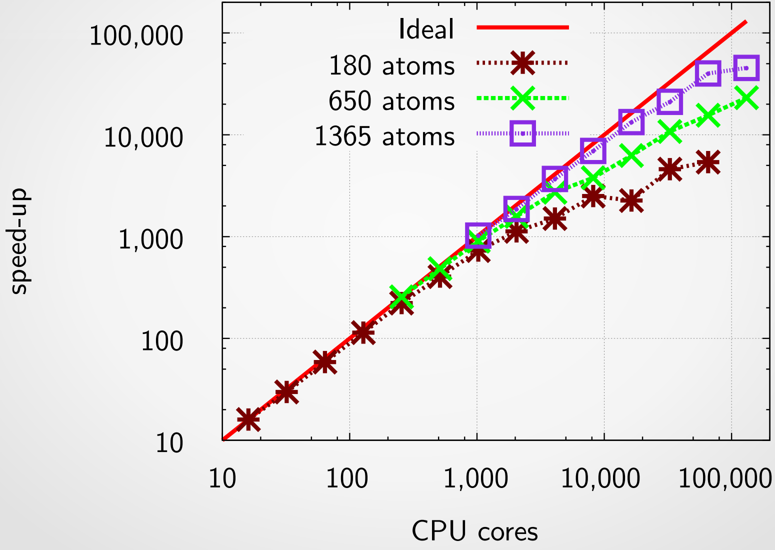
Execution-times of the PFFT solver in Genius (Blue Gene/P), Curie and Corvo (x86-64) for a system size of $L_e = 15.8$ as a function of the number of MPI processes.



TD execution-time



Speed-up of the TD



Outline

Conclusions

- FFT based methods are the most accurate ones
  - Although, all are accurate enough
- Time-dependent execution has done one step forward
  - PFFT solver has overcame the problem of the previous implementations
- FMM solver is also highly parallel, but with a big prefactor

Acknowledgements

- PRACE Research Infrastructure
- Rechenzentrum Garching (RZG) of the Max Planck Society
- European Research Council Advanced Grant DYNamo (ERC-2010-AdG-Proposal No. 267374)
- Spanish Grants (FIS2011-65702-C02-01 and PIB2010US-00652)
- ACI-Promociona (ACI2009-1036),
- General funding for research groups UPV/EHU (ALDAPA, GIU10/02)
- Grupos Consolidados UPV/EHU del Gobierno Vasco (IT-319-07)
- European Commission project CRONOS (280879-2 CRONOS CP-FP7).
- Scholarship of the University of the Basque Country UPV/EHU.

Bibliography

Joseba Alberdi-Rodriguez.
Analysis of performance and scaling of the scientific code Octopus.
LAP LAMBERT Academic Publishing, 2010.

I. Kabadshow and H. Dachsel.
The Error-Controlled Fast Multipole Method for Open and Periodic Boundary Conditions.
In *Fast Methods for Long-Range Interactions in Complex Systems*, IAS Series, Volume 6, Forschungszentrum Jülich, Germany, 2010. CECAM.

Michael Pippig.
An Efficient and Flexible Parallel FFT Implementation Based on FFTW.
In Bischof, Christian and Hegering, Heinz-Gerd and Nagel, Wolfgang E. and Wittum, Gabriel, editor, *Competence in High Performance Computing*, pages 125–134. Springer, 2010.

Implementation and measurement of new Poisson solvers in `Octopus`

Lyon
October 22-23

Joseba Alberdi-Rodriguez
University of the Basque Country UPV/EHU

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

ALDAPA
nano-bio spectroscopy group