Activity Report 2015

Project-Team ECUADOR

Program transformations for scientific computing

RESEARCH CENTER
Sophia Antipolis - Méditerranée

THEME
Numerical schemes and simulations

Table of contents

Project-Team ECUADOR

Creation of the Project-Team: 2014 January 01

Keywords:

Computer Science and Digital Science:

2.1.1. - Semantics of programming languages
2.2.1. - Static analysis
2.5. - Software engineering
6.1.1. - Continuous Modeling (PDE, ODE)
6.2.6. - Optimization
6.2.7. - High performance computing
6.3.1. - Inverse problems
6.3.2. - Data assimilation

Other Research Topics and Application Domains:

1.1.2. - Molecular biology
3.2. - Climate and meteorology
3.3.2. - Water: sea & ocean, lake & river
3.3.4. - Atmosphere
5.2.3. - Aviation
5.2.4. - Aerospace
9.5.3. - Economy, Finance

1. Members

Research Scientists

Laurent Hascoët [Team leader, Inria, Senior Researcher, HdR]
Alain Dervieux [Inria, Senior Researcher, HdR]
Valérie Pascual [Inria, Researcher]

PhD Students

Gautier Brèthes [Inria]
Eléonore Gauci [Inria]
Ala Taftaf [Inria]

Post-Doctoral Fellow

Guilherme Coelho Cunha [Inria, from Mar 2015]

Visiting Scientists

Olivier Allain [LEMMA]
Marcin Wyrozebski [Warsaw University of Technology, from Sep 2015 until Oct 2015]

Administrative Assistant

Christine Claux [Inria]

Others

Raphaël Couronné [Etudiant SUPELEC, from Jul 2015 until Sep 2015]
Bruno Koobus [Univ. Montpellier II]
Stephen Wornom [LEMMA]

2. Overall Objectives

2.1. Overall Objectives

Team Ecuador studies Algorithmic Differentiation (AD) of computer programs, blending :

- **AD theory:** We study software engineering techniques, to analyze and transform programs mechanically. Algorithmic Differentiation (AD) transforms a program `P` that computes a function F, into a program `P'` that computes analytical derivatives of F. We put emphasis on the *adjoint mode* of AD, a sophisticated transformation that yields gradients for optimization at a remarkably low cost.
- **AD application to Scientific Computing:** We adapt the strategies of Scientific Computing to take full advantage of AD. We validate our work on real-size applications.

We want to produce AD code that can compete with hand-written sensitivity and adjoint programs used in the industry. We implement our algorithms into the tool Tapenade, one of the most popular AD tools now.

Our research directions :

- Efficient adjoint AD of frequent dialects e.g. Fixed-Point loops.
- Development of the adjoint AD model towards Dynamic Memory Management.
- Development of the adjoint AD model towards Parallel Languages.
- Optimal shape design and optimal control for steady and unsteady simulations. Higher-order derivatives for uncertainty quantification.
- Adjoint-driven mesh adaptation.

3. Research Program

3.1. Algorithmic Differentiation

Participants: Laurent Hascoët, Valérie Pascual, Ala Taftaf.

algorithmic differentiation (AD, aka Automatic Differentiation) Transformation of a program, that returns a new program that computes derivatives of the initial program, i.e. some combination of the partial derivatives of the program's outputs with respect to its inputs.

adjoint Mathematical manipulation of the Partial Differential Equations that define a problem, obtaining new differential equations that define the gradient of the original problem's solution.

checkpointing General trade-off technique, used in adjoint AD, that trades duplicate execution of a part of the program to save some memory space that was used to save intermediate results.

Algorithmic Differentiation (AD) differentiates *programs*. The input of AD is a source program P that, given some $X \in \mathbb{R}^n$, returns some $Y = F(X) \in \mathbb{R}^m$, for a differentiable F. AD generates a new source program P' that, given X, computes some derivatives of F [6].

Any execution of P amounts to a sequence of instructions, which is identified with a composition of vector functions. Thus, if

$$\begin{array}{ccc} P & \text{runs} & \{I_1; I_2; \cdots I_p;\}, \\ F & \text{then is} & f_p \circ f_{p-1} \circ \cdots \circ f_1, \end{array} \tag{1}$$

where each f_k is the elementary function implemented by instruction I_k. AD applies the chain rule to obtain derivatives of F. Calling X_k the values of all variables after instruction I_k, i.e. $X_0 = X$ and $X_k = f_k(X_{k-1})$, the Jacobian of F is

$$F'(X) = f'_p(X_{p-1}) \,.\, f'_{p-1}(X_{p-2}) \,.\, \cdots \,.\, f'_1(X_0) \tag{2}$$

which can be mechanically written as a sequence of instructions I'_k. This can be generalized to higher level derivatives, Taylor series, etc. Combining the I'_k with the control of P yields P', and therefore this differentiation is piecewise.

In practice, many applications only need cheaper projections of $F'(X)$ such as:

- **Sensitivities**, defined for a given direction $\dot{X}$ in the input space as:

$$F'(X).\dot{X} = f'_p(X_{p-1}) \,.\, f'_{p-1}(X_{p-2}) \,.\, \cdots \,.\, f'_1(X_0) \,.\, \dot{X} \quad . \tag{3}$$

  This expression is easily computed from right to left, interleaved with the original program instructions. This is the *tangent mode* of AD.

- **Adjoints**, defined after transposition (F'^*), for a given weighting $\overline{Y}$ of the outputs as:

$$F'^*(X).\overline{Y} = f_1'^*(X_0).f_2'^*(X_1). \,\cdots\, .f_{p-1}'^*(X_{p-2}).f_p'^*(X_{p-1}).\overline{Y} \quad . \tag{4}$$

  This expression is most efficiently computed from right to left, because matrix$\times$vector products are cheaper than matrix$\times$matrix products. This is the *adjoint mode* of AD, most effective for optimization, data assimilation [32], adjoint problems [27], or inverse problems.

Adjoint AD builds a very efficient program [29]. This adjoint program will compute the gradient in a time independent from the number of parameters n, and which is only a small multiple of the run-time of P. In contrast, computing the same gradient with the *tangent mode* would require running the tangent differentiated program n times.

However, the X_k are required in the *inverse* of their computation order. If the original program *overwrites* a part of X_k, the differentiated program must restore X_k before it is used by $f_{k+1}'^*(X_k)$. Therefore, the central research problem of adjoint AD is to make the X_k available in reverse order at the cheapest cost, using strategies that combine storage, repeated forward computation from available previous values, or even inverted computation from available later values.

Another research issue is to make the AD model cope with the constant evolution of modern language constructs. From the old days of Fortran77, novelties include pointers and dynamic allocation, modularity, structured data types, objects, vectorial notation and parallel programming. We keep developing our models and tools to handle these new constructs.

3.2. Static Analysis and Transformation of programs

Participants: Laurent Hascoët, Valérie Pascual, Ala Taftaf.

abstract syntax tree Tree representation of a computer program, that keeps only the semantically significant information and abstracts away syntactic sugar such as indentation, parentheses, or separators.

control flow graph Representation of a procedure body as a directed graph, whose nodes, known as basic blocks, each contain a sequence of instructions and whose arrows represent all possible control jumps that can occur at run-time.

abstract interpretation Model that describes program static analysis as a special sort of execution, in which all branches of control switches are taken concurrently, and where computed values are replaced by abstract values from a given *semantic domain*. Each particular analysis gives birth to a specific semantic domain.

data flow analysis Program analysis that studies how a given property of variables evolves with execution of the program. Data Flow analysis is static, therefore studying all possible run-time behaviors and making conservative approximations. A typical data-flow analysis is to detect, at any location in the source program, whether a variable is initialized or not.

data dependence analysis Program analysis that studies the itinerary of values during program execution, from the place where a value is defined to the places where it is used, and finally to the place where it is overwritten. The collection of all these itineraries is stored as *Def-Use and Use-Def chains* or as a *data dependence graph*, and data flow analysis most often rely on this graph.

data dependence graph Directed graph that relates accesses to program variables, from the write access that defines a new value to the read accesses that use this value, and from the read accesses to the write access that overwrites this value. Dependences express a partial order between operations, that must be preserved to preserve the program's result.

The most obvious example of a program transformation tool is certainly a compiler. Other examples are program translators, that go from one language or formalism to another, or optimizers, that transform a program to make it run better. AD is just one such transformation. These tools use sophisticated analysis [20]. These tools share their technological basis. More importantly, there are common mathematical models to specify and analyze them.

An important principle is *abstraction*: the core of a compiler should not bother about syntactic details of the compiled program. The optimization and code generation phases must be independent from the particular input programming language. This is generally achieved using language-specific *front-ends*, language-independent *middle-ends*, and target-specific *back-ends*. In the middle-end, analysis can concentrate on the semantics of only a small set of constructs. This analysis operates on an abstract representation of programs made of one *call graph*, whose nodes are themselves *flow graphs* whose nodes (*basic blocks*) contain abstract *syntax trees* for the individual atomic instructions. To each level are attached symbol tables, nested to capture scoping.

Static program analysis can be defined on this internal representation, which is largely language independent. The simplest analyses on trees can be specified with inference rules [23], [30], [21]. But many *data-flow analyses* are more complex, and better defined on graphs than on trees. Since both call graphs and flow graphs may be cyclic, these global analyses will be solved iteratively. *Abstract Interpretation* [24] is a theoretical framework to study complexity and termination of these analyses.

Data flow analyses must be carefully designed to avoid or control combinatorial explosion. At the call graph level, they can run bottom-up or top-down, and they yield more accurate results when they take into account the different call sites of each procedure, which is called *context sensitivity*. At the flow graph level, they can run forwards or backwards, and yield more accurate results when they take into account only the possible execution flows resulting from possible control, which is called *flow sensitivity*.

Even then, data flow analyses are limited, because they are static and thus have very little knowledge of actual run-time values. Far before reaching the very theoretical limit of *undecidability*, one reaches practical limitations to how much information one can infer from programs that use arrays [36], [25] or pointers. Therefore, conservative *over-approximations* must be made, leading to derivative code less efficient than ideal.

3.3. Algorithmic Differentiation and Scientific Computing

Participants: Alain Dervieux, Laurent Hascoët, Bruno Koobus.

linearization In Scientific Computing, the mathematical model often consists of Partial Differential Equations, that are discretized and then solved by a computer program. Linearization of these equations, or alternatively linearization of the computer program, predict the behavior of the model when small perturbations are applied. This is useful when the perturbations are effectively small, as in acoustics, or when one wants the sensitivity of the system with respect to one parameter, as in optimization.

adjoint state Consider a system of Partial Differential Equations that define some characteristics of a system with respect to some input parameters. Consider one particular scalar characteristic. Its sensitivity, (or gradient) with respect to the input parameters can be defined as the solution of "adjoint" equations, deduced from the original equations through linearization and transposition. The solution of the adjoint equations is known as the adjoint state.

Scientific Computing provides reliable simulations of complex systems. For example it is possible to *simulate* the steady or unsteady 3D air flow around a plane that captures the physical phenomena of shocks and turbulence. Next comes *optimization*, one degree higher in complexity because it repeatedly simulates and applies gradient-based optimization steps until an optimum is reached. The next sophistication is *robustness* i.e. to detect and to lower preference to a solution which, although maybe optimal, is very sensitive to uncertainty on design parameters or on manufacturing tolerances. This makes second derivative come into play. Similarly *Uncertainty Quantification* can use second derivatives to evaluate how uncertainty on the simulation inputs imply uncertainty on its outputs.

We investigate several approaches to obtain the gradient, between two extremes:

- One can write an *adjoint system* of mathematical equations, then discretize it and program it by hand. This is time consuming. Although this looks mathematically sound [27], this does not provide the gradient of the discretized function itself, thus degrading the final convergence of gradient-descent optimization.
- One can apply adjoint AD (*cf* 3.1) on the program that discretizes and solves the direct system. This gives exactly the adjoint of the discrete function computed by the program. Theoretical results [26] guarantee convergence of these derivatives when the direct program converges. This approach is highly mechanizable, but leads to massive use of storage and may require code transformation by hand [31], [34] to reduce memory usage.

If for instance the model is steady, or when the computation uses a Fixed-Point iteration, tradeoffs exist between these two extremes [28], [22] that combine low storage consumption with possible automated adjoint generation. We advocate incorporating them into the AD model and into the AD tools.

4. Application Domains

4.1. Algorithmic Differentiation

Algorithmic Differentiation of programs gives sensitivities or gradients, useful for instance for :

- optimum shape design under constraints, multidisciplinary optimization, and more generally any algorithm based on local linearization,
- inverse problems, such as parameter estimation and in particular 4Dvar data assimilation in climate sciences (meteorology, oceanography),
- first-order linearization of complex systems, or higher-order simulations, yielding reduced models for simulation of complex systems around a given state,
- mesh adaptation and mesh optimization with gradients or adjoints,
- equation solving with the Newton method,
- sensitivity analysis, propagation of truncation errors.

4.2. Multidisciplinary optimization

A CFD program computes the flow around a shape, starting from a number of inputs that define the shape and other parameters. On this flow one can define optimization criteria e.g. the lift of an aircraft. To optimize a criterion by a gradient descent, one needs the gradient of the output criterion with respect to all the inputs, and possibly additional gradients when there are constraints. Adjoint AD is the most efficient way to compute these gradients.

4.3. Inverse problems and Data Assimilation

Inverse problems aim at estimating the value of hidden parameters from other measurable values, that depend on the hidden parameters through a system of equations. For example, the hidden parameter might be the shape of the ocean floor, and the measurable values of the altitude and velocities of the surface.

One particular case of inverse problems is *data assimilation* [32] in weather forecasting or in oceanography. The quality of the initial state of the simulation conditions the quality of the prediction. But this initial state is not well known. Only some measurements at arbitrary places and times are available. A good initial state is found by solving a least squares problem between the measurements and a guessed initial state which itself must verify the equations of meteorology. This boils down to solving an adjoint problem, which can be done though AD [35]. Figure 1 shows an example of a data assimilation exercise using the oceanography code OPA [33] and its AD-adjoint produced by Tapenade.

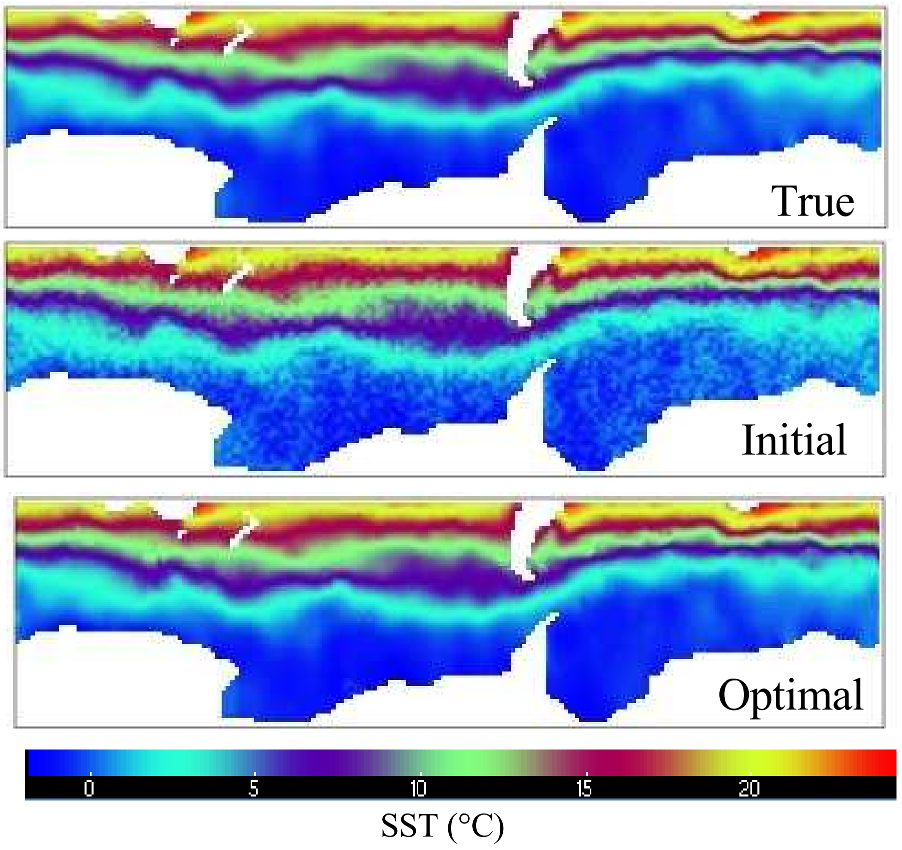

Figure 1. Twin experiment using the adjoint of OPA. Random noise, added to a simulation of the sea surface temperature around the Antarctic, is removed by minimizing the discrepancy with the physical model

The special case of *4Dvar* data assimilation is particularly challenging. The 4th dimension in “4D” is time, as available measurements are distributed over a given assimilation period. Therefore the least squares mechanism must be applied to a simulation over time that follows the time evolution model. This process gives a much better estimation of the initial state, because both position and time of measurements are taken into account. On the other hand, the adjoint problem involved is more complex, because it must run (backwards) over many time steps. This demanding application of AD justifies our efforts in reducing the runtime and memory costs of AD adjoint codes.

4.4. Linearization

Simulating a complex system often requires solving a system of Partial Differential Equations. This can be too expensive, in particular in the context of real time. When one wants to simulate the reaction of this complex system to small perturbations around a fixed set of parameters, there is an efficient approximation: just suppose that the system is linear in a small neighborhood of the current set of parameters. The reaction of the system is thus approximated by a simple product of the variation of the parameters with the Jacobian matrix of the system. This Jacobian matrix can be obtained by AD. This is especially cheap when the Jacobian matrix is sparse. The simulation can be improved further by introducing higher-order derivatives, such as Taylor expansions, which can also be computed through AD. The result is often called a *reduced model*.

4.5. Mesh adaptation

Some approximation errors can be expressed by an adjoint state. Mesh adaptation can benefit from this. The classical optimization step can give an optimization direction not only for the control parameters, but also for the approximation parameters, and in particular the mesh geometry. The ultimate goal is to obtain optimal control parameters up to a precision prescribed in advance.

5. New Software and Platforms

5.1. AIRONUM

SCIENTIFIC DESCRIPTION
Aironum is an experimental software that solves the unsteady compressible Navier-Stokes equations with k-, LES-VMS and hybrid turbulence modelling on parallel platforms, using MPI. The mesh model is unstructured tetrahedrization, with possible mesh motion.
FUNCTIONAL DESCRIPTION
Aironum was developed by Inria and University of Montpellier. It is used by Inria, University of Montpellier and University of Pisa (I). Aironum is used as an experimental platform for:

- Numerical approximation of compressible flows, such as upwind mixed element volume approximation with superconvergence on regular meshes.
- Numerical solution algorithms for the implicit time advancing of the compressible Navier-Stokes equations, such as parallel scalable deflated additive Schwarz algorithms.
- Turbulence modelling such as the Variational Multiscale Large eddy Simulation and its hybridization with RANS statistical models.
- Participant: Alain Dervieux
- Contact: Alain Dervieux
- URL: http://www-sop.inria.fr/tropics/aironum

5.2. TAPENADE

KEYWORDS: Static analysis - Optimization - Compilation - Gradients
SCIENTIFIC DESCRIPTION

Tapenade implements the results of our research about models and static analyses for AD. For a full specification and description, see [10]. AD produces analytical derivatives, that are exact up to machine precision. Adjoint AD computes gradients at a cost which is independent from the number of input variables. Tapenade performs sophisticated flow- and context-sensitive data-flow analysis on the complete source program to produce an efficient differentiated code. Analyses include Type-Checking, Read-Write analysis, Pointer analysis. AD-specific analyses include:

- Activity analysis: Detects variables whose derivative is either null or useless, to reduce the number of derivative instructions.
- Adjoint Liveness analysis: Detects the source statements that are dead code for the computation of derivatives.
- TBR analysis: In Adjoint AD, reduces the set of source variables that need to be recovered.

FUNCTIONAL DESCRIPTION
Tapenade transforms an original program into a new program that computes derivatives of the original program. Tapenade accepts source programs written in Fortran77, Fortran90, or C. Tapenade can differentiate in tangent, vector tangent, adjoint, and vector adjoint modes. Tapenade can be downloaded and installed on most architectures. Alternatively, it can be used as a web server. Higher-order derivatives can be obtained through repeated application.

- Participants: Laurent Hascoët, Valérie Pascual, Ala Taftaf
- Contact: Laurent Hascoët
- URL: http://www-sop.inria.fr/tropics/tapenade.html

6. New Results

6.1. AD-adjoints and C dynamic memory management

Participants: Laurent Hascoët, Raphaël Couronné, Sri Hari Krishna Narayanan [Argonne National Lab. (Illinois, USA)], Mathieu Morlighem [University of California at Irvine (USA)].

One of the current frontiers of AD research is the definition of an adjoint AD model that can cope with dynamic memory management. This research is central in our ongoing effort towards adjoint AD of C, and more remotely towards AD of C++. This research is conducted in collaboration with the MCS department of Argonne National Lab. Our partnership is formalized by joint participation in the Inria joint lab JLESC, and partly funded by the Partner University Fund (PUF) of the French embassy in the USA.

Adjoint AD must reproduce in reversed order the control decisions of the original code. In languages such as C, allocation of dynamic memory and pointer management form a significant part of these control decisions. Reproducing memory (de)allocation in reverse means reallocating memory, possibly receiving a different memory chunk. Reproducing pointer addresses in reverse thus require to convert addresses in the former memory chunks into equivalent addresses in the new reallocated chunks. Together with Krishna Narayanan from Argonne, we experiment on real applications to find the most efficient solution to this address conversion problem. We jointly develop a library (called ADMM, ADjoint Memory Management) whose primitives can be used in AD adjoint code to handle this address conversion. Using this library together with Tapenade, we could obtain a correct AD adjoint code of a medium-size industrial code (“Multibody", structural mechanics) that exhibits a typical usage of C pointer arithmetic. This year, the same effort was conducted with the OpenAD AD tool, leading us to an ADMM library less dependent on the particular target AD tool. A joint publication with our colleagues from Argonne is in preparation.

In parallel, we investigate alternative implementation strategies for ADMM, one of which could be to build our own memory (de)allocation mechanism, This should ultimately rely on the standard C library. As a result, management of adjoint memory addresses could be done deeper in the system and therefore with a smaller overhead, at the cost of some additional portability issues.

We pursue our objective of improving reliability of the AD adjoint model for C codes to a similar level as achieved for Fortran. To this end we apply Tapenade to increasingly larger and complex C codes. In addition to the already mentioned “Multibody" application, we initiated differentiation of two new complex applications:

- “BLN" is a code developed by the Inria team ABS, that computes the potential energy of possible conformations of a macromolecule. Its gradient is used to explore the local minima in the energy landscape of these conformations. The AD adjoint of a Fortran implementation of BLN has been built by Tapenade and successfully validated. The adjoint of the C implementation is a challenge that helps us clarify the adjoint AD model that we use in Tapenade. The C version of BLN that we are considering is actually a (partly mechanical) translation of the actual C++ source. This makes this code an even more appealing and challenging test case. This work was mostly conducted by Raphaël Couronné as a part of his summer internship with us.
- “SEISM" is a code developed by Mathieu Morlighem from UC Irvine, jointly with Eric Larour from JPL. This is a glaciology code closely related to the larger “ISSM" code, in C++. One objective, addressed mostly by Mathieu Morlighem, is to clarify recommendations on the C programming style (again very much inspired here from the C++ style) that allows AD to perform better. The other objective, adressed mostly by our team, it to experiment with quite intricate data structures, where Tapenade’s static pointer destination analysis is used intensively.

6.2. AD-adjoints of MPI-parallel codes

Participants: Laurent Hascoët, Ala Taftaf, Sri Hari Krishna Narayanan [Argonne National Lab. (Illinois, USA)].

We have a long-standing collaboration with Argonne National Lab on the question of adjoint AD of message-passing parallel codes. We continued joint development of the Adjoinable-MPI library (AMPI) that provides efficient tangent and adjoint AD for MPI-parallel codes, independently of the AD tool used (now AdolC, dco, OpenAD, Tapenade).

During her PhD work, Ala Taftaf is considering the question of checkpointing applied to the AD-adjoint of an MPI-parallel code. Checkpointing is a memory/runtime tradeoff which is essential for adjoint AD of large codes, in particular parallel codes. However, for MPI codes this question has always been addressed by ad-hoc hand manipulations of the differentiated code, and with no formal assurance of correctness. Ala Taftaf is investigating the assumptions implicitly made during past experiments, to clarify and generalize them. On one hand we propose an extension of the adjoint of MPI point-to-point communication primitives, so that the semantics of an adjoint program is preserved for any placement of checkpoints. On the other hand, we propose an alternative extension of these adjoint communications, more efficient but that requires a number of restrictions on the placement of checkpoints. We shall try to provide proof of correctness of these strategies, and in particular demonstrate that they cannot introduce deadlocks. Tradeoffs between the two extensions should be investigated. Ala Taftaf presented her research on “‘Adjoint-Checkpointing on MPI-parallel codes” at the EuroAD workshop in Paderborn, Germany, december 1-2. A conference article has been submitted to Eccomas 2016 in Crete.

6.3. AD-adjoints of Iterative Processes

Participants: Laurent Hascoët, Ala Taftaf, Sri Hari Krishna Narayanan [Argonne National Lab. (Illinois, USA)], Daniel Goldberg [University of Edinburgh, UK].

Adjoint codes naturally propagate partial gradients backwards from the result of the simulation. However, this uses the data flow of the simulation in reverse order, at a cost that increases with the length of the simulation. In the special case of iterative Fixed-Point loops, only the final converged result should be used: the “initial guess” and the intermediate non-converged states should not be considered by the adjoint calculation, and this remark brings enormous gain in memory use. We selected the strategy proposed by Bruce Christianson [22] and this year we continued its application to medium-size testcases provided by Queen Mary University for the AboutFlow project. We also simplified the user interface provided to trigger this special strategy extension in Tapenade. Ala Taftaf presented her results at the ECCOMAS Eurogen conference in Glasgow [15], september 14-16.

In parallel, we collaborated with Krishna Narayanan from ANL and Dan Goldberg from University of Edinburgh (UK) to implement the same strategy into the OpenAD tool, in view of applying it to a glaciology configuration of the MIT GCM code. A joint article describing the results has been submitted for publication.

6.4. AD-adjoints of large real codes

Participants: Laurent Hascoët, Valérie Pascual, Raphaël Couronné, Fabrice Zaoui [EDF R&D, LNHE].

In collaboration with EDF, Valérie Pascual is applying Tapenade to the hydrographic code “Mascaret". Both tangent and adjoint diferentiated codes have been built and validated. Application of the tangent differentiated Mascaret for data assimilation on two real cases is described in a joint publication [14].

During his summer internship, Raphaël Couronné has applied Tapenade to the MIT “GCM", a reference code in the Earth Sciences community. We have obtained a valid adjoint for a recommended configuration of this very large Fortran code. This test showed some maturity of the Tapenade tool for Fortran, as it turned out that no modification nor debug of the tool was needed. We are now discussing with the MIT team to schedule further collaboration.

In cooperation with the partners of the FP7 project UMRIDA, the team has assisted Alenia-Aermacchi (Filomena Cariglino and Nicola Ceresola) in the efficient differentiation of their Euler/Navier Stokes code “UNS3D" in tangent mode, dealing in particular with its use of MPI.

The team has assisted Marcin Wyrozebski from Warsaw University of Technology, to apply Tapenade to a CFD software from WUT.

6.5. Resolution of linearised systems and efficiency

Participants: Olivier Allain [Lemma], Gautier Brèthes, Alain Dervieux, Bruno Koobus [Université Montpellier 2], Emmanuelle Itam [Université Montpellier 2], Vincent Levasseur [Lemma], Stephen Wornom [Lemma].

For Fluid Mechanics as well as for Structural Mechanics, an implicit time-advancing is mandatory. It can be applied efficiently if the large systems involved are solved with a good parallel algorithm. In the 90’s, a generation of solution algorithms was devised on the basis of Domain Decomposition Methods (DDM). For complex models (compressible flows...), Schwarz DDM were combined with quasi-Newton algorithms such as GMRES. These are for example Restrictive Additive Schwarz (RAS), which is used in our platform AIRONUM. RAS was developed by Cai, Farhat and others. RAS is an ancestor of the widely used class of Newton-Krylov-Schwarz (NKS) algorithms. For hundreds of processors many versions of NKS, and in particular RAS, are almost scalable (convergence rate independant of the number of processors). But scalability vanishes for a medium-large number of processors (thousands). In the ANR ECINADS, coordinated by Ecuador, a Coarse-Grid Deflated RAS was developped: iteration-wise scalability holds for all parts, except for the coarse grid direct solver, which concerns a much smaller problem. Effective Convergence Scalability (ECS) was confirmed up to 2048 processors. Beyond this level the asymptotic complexity of the coarse-grid direct solver becomes predominant and ECS is lost. In other words, with a Coarse-Grid Deflated RAS, the size of the coarse grid problem which is solved by a direct algebraic solver must be limited in order to enjoy ECS. For finer meshes, the coarse system cannot be finer, and the efficiency is lower. It is then natural to consider intermediate meshes on which iterative solvers will be applied. In the ANR MAIDESC, Gautier Brèthes has defined a multi-mesh Full MultiGrid (FMG) algorithm adapted to anisotropic mesh adaptation. In 2015, the method has been extended to MPI-based massive parallelism, in cooperation with the Lemma team for the computation of incompressible flows. As a perspective, our parallel MG can be complemented with the previous version of the solver (deflated RAS) for a higher degree of scalability.

A second issue which we addressed is the use of explicit time advancing. Many unsteady flows have to be computed with explicit time advancing. A single explicit time step is of a low cost and can be highly accurate. Explicit time advancing is mandatory for wave propagation: blast shocks of vortices in wakes. However the meshes used may involve small regions in which the explicit time step should be very small and large regions in which such a small time step is a waste. The family of time-advancing methods in which unsteady phenomena are computed using different time steps in different regions is called the multirate methods. In our cooperation with University of Montpellier, a novel multirate method using cell-agglomeration has been designed and developed in our AIRONUM platform. An article is in preparation. This work takes place in the ANR MAIDESC programme.

6.6. Control of approximation errors

Participants: Gautier Brèthes, Eléonore Gauci, Alain Dervieux, Adrien Loseille [GAMMA team, Inria-Rocquencourt], Frédéric Alauzet [GAMMA team, Inria-Rocquencourt], Stephen Wornom [Lemma], Anca Belme [university of Paris 6].

The study of combination of full multigrid (FMG) with anisotropic mesh adaption (AA), started with the thesis of Gautier Brèthes, has been published [13].

Further studies of mesh adaptation for viscous flows are currently performed and a journal paper, joint with Inria team Gamma3 and University of Paris 6 (Anca Belme) is in preparation.

An important novelty in mesh adaption is the norm-oriented AA method. The method relies on the definition of ad hoc correctors. It has been developed in the academic platform “FMG" for elliptic problems. Gautier Brèthes gave several presentations in conferences, a journal article has been submitted. The introduction of the norm-oriented idea considerably amplifies the impact of adjoint-based AA. The applied mathematician and the engineer now have methods when faced to mesh adaptation for the simulation of a complex PDE system, since they can specify which error norm level they wish, and for which norm [12], [16]. Another version is developed jointly with Inria team Gamma3 for the compressible Euler model [19].

A cooperation has started between Gautier Brèthes et Thierry Coupez (Ecole Centrale de Nantes) concerning discrete metrics. This takes place in the ANR MAIDESC program. An article is in preparation.

Éléonore Gauci started last year a thesis (co-advised by Frédéric Alauzet) on the study of norm-oriented criteria for CFD and coupled CSM-CFD systems. She gave a presentation at the “Coupled Problems" symposium.

Post-doc Guilherme Cunha did a study (in cooperation with Lemma) on the combination of mesh adaptation and coefficient identification for unsteady phenomena.

The theoretical studies are supported by an ANR project MAIDESC coordinated by ECUADOR and Gamma3, which deals with meshes for interfaces, third-order accuracy, meshes for boundary layers, and curved meshes.

CFD application are supported by the European FP7 project UMRIDA which deals with the application of AA to approximation error modelling and control.

6.7. Turbulence models

Participants: Alain Dervieux, Bruno Koobus [University of Montpellier 2], Emmanuelle Itam [University of Montpellier 2], Marianna Braza [CNRS-IMFT at Toulouse], Stephen Wornom [Lemma], Bruno Sainte-Rose [Lemma].

The purpose of our work in hybrid RANS/LES is to develop new approaches for industrial applications of LES-based analyses. In the applications targetted (aeronautics, hydraulics), the Reynolds number can be as high as several tenth millions, far too high for pure LES models. However, certain regions in the flow can be better predicted with LES than with usual statistical RANS (Reynolds averaged Navier-Stokes) models. These are mainly vortical separated regions as assumed in one of the most popular hybrid model, the hybrid Detached Eddy Simulation model. Here, “hybrid” means that a blending is applied between LES and RANS. An important difference between a real life flow and a wind tunnel or basin is that the turbulence of the flow upstream of each body is not well known.

This year, we have continued the evaluation of a dynamic formulation of Piomelli-Germano type for the Variational-multiscale model. We have also modified the integration of the boundary layer by adding the so-called Menter correction imposing the Bradshaw law. We have studied these improvements on multiple-body flows. An emblematic case is the interaction between two parallel cylinders, one being in the wake of the other. A flow around a space probe at high Reynolds number is also studied [18], [17].

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

- Ecuador and Lemma share the results of Gautier Brèthes' thesis, which is partly supported by Lemma, the other part being supported by a PACA region fellowship.
- Ecuador and Lemma have a bilateral contract to share the results of Stephen Wornom, Lemma engineer provided to Inria and hosted by Inria under a Inria-Lemma contract.
- Ecuador and EDF have a bilateral contract on AD of the hydrology code "Mascaret". The correspondent on the Ecuador side is Valérie Pascual.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

8.1.1.1. MAIDESC

Ecuador is coordinator of the ANR project MAIDESC, with Inria team Gamma3, University of Montpellier II, CEMEF-Ecole des Mines, Inria-Bordeaux, Lemma and Transvalor. MAIDESC concentrates on mesh adaptation and in particular meshes for interfaces, third-order accuracy, meshes for boundary layers, and curved meshes.

8.2. European Initiatives

8.2.1. FP7 & H2020 Projects

8.2.1.1. AboutFlow

Type: PEOPLE

Instrument: Initial Training Network

Duration: 2012-2016

Coordinator: Jens-Dominik Mueller

Partner: Queen Mary University of London (UK)

Inria contact: Laurent Hascoët

Abstract: The aim of AboutFlow is to develop robust gradient-based optimisation methods using adjoint sensitivities for numerical optimisation of flows. http://aboutflow.sems.qmul.ac.uk/

8.2.1.2. UMRIDA

Type:AAT

Instrument:Aeronautics and Air Transport

Duration: 2013-2016

Coordinator: Charles Hirsch

Partner: Numeca S.A. (Belgium)

Inria contact: Alain Dervieux

Abstract: UMRIDA addresses major research challenges in Uncertainty Quantification and Robust Design: develop new methods that handle large numbers of simultaneous uncertainties and generalized geometrical uncertainties. Apply these methods to representative industrial configurations.

8.3. International Initiatives

8.3.1. Inria International Labs

Ecuador participates in the Joint Laboratory for Exascale Computing (JLESC) together with colleagues at Argonne National Laboratory. Laurent Hascoët attended the JLESC meeting in Bonn, Germany, december 2-5.

8.4. International Research Visitors

8.4.1. Visits of International Scientists

- Krishna Narayanan from Argonne National Laboratory, september 21-25.

8.4.2. Internships

- Marcin Wyrozebski from Warsaw University of Technology, september 1-30.

8.4.3. Visits to International Teams

- Laurent Hascoët visited Argonne National Laboratory, april 13-23.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific events organisation

9.1.1.1. Member of the organizing committees

- Laurent Hascoët is on the organizing commitee of the EuroAD Workshops on Algorithmic Differentiation.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Master : Laurent Hascoët, Optimisation avancée, 15 h, M2, University of Nice

9.2.2. Supervision

PhD : Gautier Brèthes, “Multigrilles anisotropes adaptatives”, defended december 10, advisor A. Dervieux

PhD in progress : Ala Taftaf, “Adjoint Automatic Differentiation on High-performance codes”, started july 2013, advisor L. Hascoët.

PhD in progress : Éléonore Gauci, “Norm-oriented criteria for CFD and coupled CSM-CFD systems”, started october 2014, advisor A. Dervieux

9.2.3. Juries

- Alain Dervieux, jury, PhD defense of Nicolas Barral, University Paris VI, november 27.
- Alain Dervieux, jury, PhD defense of Vilas Schinde, X-IMSIA, december 17.

9.3. Popularization

Alain Dervieux and Ala Taftaf participated to the event “la fête de la science” in Antibes, october 10-11.

10. Bibliography

Major publications by the team in recent years

[1] F. COURTY, A. DERVIEUX. *Multilevel functional Preconditioning for shape optimisation*, in "International Journal of CFD", 2006, vol. 20, n^{o} 7, pp. 481-490

[2] F. COURTY, A. DERVIEUX, B. KOOBUS, L. HASCOËT. *Reverse automatic differentiation for optimum design: from adjoint state assembly to gradient computation*, in "Optimization Methods and Software", 2003, vol. 18, n^{o} 5, pp. 615-627

[3] B. DAUVERGNE, L. HASCOËT. *The Data-Flow Equations of Checkpointing in reverse Automatic Differentiation*, in "International Conference on Computational Science, ICCS 2006, Reading, UK", 2006

[4] L. HASCOËT, M. ARAYA-POLO. *The Adjoint Data-Flow Analyses: Formalization, Properties, and Applications*, in "Automatic Differentiation: Applications, Theory, and Tools", H. M. BÜCKER, G. CORLISS, P. HOVLAND, U. NAUMANN, B. NORRIS (editors), Lecture Notes in Computational Science and Engineering, Springer, 2005

[5] L. HASCOËT, S. FIDANOVA, C. HELD. *Adjoining Independent Computations*, in "Automatic Differentiation of Algorithms: From Simulation to Optimization", New York, NY, G. CORLISS, C. FAURE, A. GRIEWANK, L. HASCOËT, U. NAUMANN (editors), Computer and Information Science, Springer, 2001, chap. 35, pp. 299-304

[6] L. HASCOËT. *Adjoints by Automatic Differentiation*, in "Advanced data assimilation for geosciences", Oxford University Press, 2014, https://hal.inria.fr/hal-01109881

[7] L. HASCOËT, U. NAUMANN, V. PASCUAL. *"To Be Recorded" Analysis in Reverse-Mode Automatic Differentiation*, in "Future Generation Computer Systems", 2004, vol. 21, n^{o} 8

[8] L. HASCOËT, J. UTKE, U. NAUMANN. *Cheaper Adjoints by Reversing Address Computations*, in "Scientific Programming", 2008, vol. 16, n^{o} 1, pp. 81–92

[9] L. HASCOËT, M. VÁZQUEZ, B. KOOBUS, A. DERVIEUX. *A Framework for Adjoint-based Shape Design and Error Control*, in "Computational Fluid Dynamics Journal", 2008, vol. 16, n^{o} 4, pp. 454-464

[10] L. HASCOËT, V. PASCUAL. *The Tapenade Automatic Differentiation tool: Principles, Model, and Specification*, in "ACM Transactions On Mathematical Software", 2013, vol. 39, n^{o} 3, http://dx.doi.org/10.1145/2450153.2450158

[11] M. VÁZQUEZ, A. DERVIEUX, B. KOOBUS. *Multilevel optimization of a supersonic aircraft*, in "Finite Elements in Analysis and Design", 2004, vol. 40, pp. 2101-2124

Publications of the year

Doctoral Dissertations and Habilitation Theses

[12] G. BRETHES. *Adaptative and scalable mesh adaptation*, Université Nice Sophia Antipolis, December 2015, https://hal.inria.fr/tel-01256185

Articles in International Peer-Reviewed Journals

[13] G. BRETHES, O. ALLAIN, A. DERVIEUX. *A Mesh-Adaptive Metric-Based Full-Multigrid for the Poisson problem*, in "Internationale journal for numerical methods in fluids", June 2015, https://hal.inria.fr/hal-01255500

[14] F. DEMANGEON, C. GOEURY, F. ZAOUI, N. GOUTAL, V. PASCUAL, L. HASCOËT. *Algorithmic differentiation applied to the optimal calibration of a shallow water model*, in "La Houille Blanche - Revue Internationale de l'Eau", 2015, https://hal.inria.fr/hal-01244264

International Conferences with Proceedings

[15] A. TAFTAF, L. HASCOËT, V. PASCUAL. *Implementation and measurements of an efficient Fixed Point Adjoint*, in "EUROGEN 2015", GLASGOW, United Kingdom, ECCOMAS, September 2015, https://hal.inria.fr/hal-01244298

Conferences without Proceedings

[16] G. BRETHES, A. LOSEILLE, F. ALAUZET, A. DERVIEUX. *Estimates- and Corrector-based Mesh Adaptation*, in "1st Pan-American Congress on Computational Mechanics - PANACM 2015 XI Argentine Congress on Computational Mechanics - MECOM 2015", Buenos Aires, Argentina, April 2015, https://hal.inria.fr/hal-01256111

[17] E. ITAM, S. WORNOM, B. KOOBUS, A. DERVIEUX. *Application of a Hybrid Variational Multiscale Model to Massively Separated Flows*, in "3AF", Toulouse, France, March 2015, https://hal.inria.fr/hal-01256081

[18] E. ITAM, S. WORNOM, B. KOOBUS, B. SAINTE-ROSE, A. DERVIEUX. *Simulation of multiple blunt-body flows with a hybrid variational multiscale model*, in "Conference on Modelling Fluid Flow (CMFF'15)", Budapest, Hungary, September 2015, https://hal.inria.fr/hal-01256084

[19] A. LOSEILLE, A. DERVIEUX, F. ALAUZET. *Anisotropic Norm-Oriented Mesh Adaptation for Compressible Inviscid Flows*, in "53rd AIAA Aerospace Sciences Meeting, AIAA SciTech", Kissimmee, Florida, United States, January 2015, https://hal.inria.fr/hal-01256131

References in notes

[20] A. AHO, R. SETHI, J. ULLMAN. *Compilers: Principles, Techniques and Tools*, Addison-Wesley, 1986

[21] I. ATTALI, V. PASCUAL, C. ROUDET. *A language and an integrated environment for program transformations*, Inria, 1997, n^{o} 3313, http://hal.inria.fr/inria-00073376

[22] B. CHRISTIANSON. *Reverse accumulation and implicit functions*, in "Optimization Methods and Software", 1998, vol. 9, n^{o} 4, pp. 307–322

[23] D. CLÉMENT, J. DESPEYROUX, L. HASCOËT, G. KAHN. *Natural semantics on the computer*, in "Proceedings, France-Japan AI and CS Symposium, ICOT", 1986, pp. 49-89, Also, Information Processing Society of Japan, Technical Memorandum PL-86-6. Also Inria research report # 416, http://hal.inria.fr/inria-00076140

[24] P. COUSOT. *Abstract Interpretation*, in "ACM Computing Surveys", 1996, vol. 28, n^{o} 1, pp. 324-328

[25] B. CREUSILLET, F. IRIGOIN. *Interprocedural Array Region Analyses*, in "International Journal of Parallel Programming", 1996, vol. 24, n^{o} 6, pp. 513–546

[26] J. GILBERT. *Automatic differentiation and iterative processes*, in "Optimization Methods and Software", 1992, vol. 1, pp. 13–21

[27] M.-B. GILES. *Adjoint methods for aeronautical design*, in "Proceedings of the ECCOMAS CFD Conference", 2001

[28] A. GRIEWANK, C. FAURE. *Reduced Gradients and Hessians from Fixed Point Iteration for State Equations*, in "Numerical Algorithms", 2002, vol. 30(2), pp. 113–139

[29] A. GRIEWANK, A. WALTHER. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, 2nd, SIAM, Other Titles in Applied Mathematics, 2008

[30] L. HASCOËT. *Transformations automatiques de spécifications sémantiques: application: Un vérificateur de types incremental*, Université de Nice Sophia-Antipolis, 1987

[31] P. HOVLAND, B. MOHAMMADI, C. BISCHOF. *Automatic Differentiation of Navier-Stokes computations*, Argonne National Laboratory, 1997, n^{o} MCS-P687-0997

[32] F.-X. LE DIMET, O. TALAGRAND. *Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects*, in "Tellus", 1986, vol. 38A, pp. 97-110

[33] G. MADEC, P. DELECLUSE, M. IMBARD, C. LEVY. *OPA8.1 ocean general circulation model reference manual*, Pole de Modelisation, IPSL, 1998

[34] B. MOHAMMADI. *Practical application to fluid flows of automatic differentiation for design problems*, in "Von Karman Lecture Series", 1997

[35] N. ROSTAING. *Différentiation Automatique: application à un problème d'optimisation en météorologie*, université de Nice Sophia-Antipolis, 1993

[36] R. RUGINA, M. RINARD. *Symbolic Bounds Analysis of Pointers, Array Indices, and Accessed Memory Regions*, in "Proceedings of the ACM SIGPLAN'00 Conference on Programming Language Design and Implementation", ACM, 2000