INRIA, Evaluation of Theme Num

Project-team TROPICS

march 2009

Project-team title : TROPICS

Scientific leader : Laurent Hascoët

Research center : Sophia-Antipolis

1 Personnel

Personnel (march 2005)

	Misc.	INRIA	CNRS	University	**Total**
DR (1) / Professors		1			**1**
CR (2) / Assistant Professors		2			**2**
Permanent Engineers (3)		1			**1**
Temporary Engineers (4)		1			**1**
PhD Students		2			**2**
Post-Doc.					
Total		**7**			**7**
External Collaborators	1			1	**2**
Visitors ($>$ 1 month)					

(1) "Senior Research Scientist (Directeur de Recherche)"
(2) "Junior Research Scientist (Chargé de Recherche)"
(3) "Civil servant (CNRS, INRIA, ...)"
(4) "Associated with a contract (Ingénieur Expert or Ingénieur Associé)"

Personnel (march 2009)

	Misc.	INRIA	CNRS	University	**Total**
DR / Professors		2			**2**
CR / Assistant Professor		1			**1**
Permanent Engineer					
Temporary Engineer					
PhD Students		1			**1**
Post-Doc.					
Total		**4**			**4**
External Collaborators	1			1	**2**
Visitors ($>$ 1 month)					

Changes in staff

DR / Professors CR / Assistant Professors	Misc.	INRIA	CNRS	University	total
Arrival					
Leaving					

Current composition of the project-team (march 2009):

- Laurent Hascoët, DR
- Alain Dervieux, DR
- Valérie Pascual, CR
- Anca Belme, PhD student

Current position of former project-team members (including PhD students during the (put here the evaluation period) period):

- Mauricio Araya-Polo (was PhD student): Researcher, Barcelona Computing Center, Spain, `http://www.bsc.es`
- Benjamin Dauvergne (was PhD student): Engineer, "Entrouvert" company, Jouy-en-Josas, France, `http://www.entrouvert.com/fr`
- Hicham Tber (was Post-Doc): Post-Doc, IFB research team (Michael Hintermüller) U. Graz, Austria, `http://www.uni-graz.at/imawww/ifb`
- Massimiliano Martinelli (was PhD student and Post-Doc): Maths Engineer, T.E.A. group, Milano, Italy, `http://www.teasistemi.com`
- Christophe Massol (was Development Engineer): Engineer, Sogeti High Tech, Sophia Antipolis, France, `http://www.fr.sogeti.com/FR/high_tech/index.aspx`

Last INRIA enlistments

- Laurent Hascoët was promoted to DR2 position in September 2006.

2 Work progress

2.1 Keywords

Automatic Differentiation, program analysis, data-flow analysis, program transformation, scientific computing, gradient-based optimization

2.2 Context and overall goal of the project

Automatic Differentiation (AD) is one way to obtain the analytical derivatives of a mathematical function when this function is provided not as a mathematical formula but rather as a computer program.

Taking the approach known as "program transformation", AD is akin to a sophisticated form of compiling: AD reads in and analyzes the given computer program and produces a new computer program that computes the derivatives.

A prominent application field of AD in Scientific Computing is the computation of gradients of functionals, for optimization or inverse problems. Therefore the so-called "reverse mode" of AD, which is the most efficient way of computing gradients, deserves most of our attention.

It turns out that the reverse mode of AD raises Computer Science questions vastly more complex than the basic "tangent mode" of AD. Reverse AD actually needs data-flow analysis of the same power as for optimizing or parallelizing compilers. It also requires analysis that have no equivalent in compilers. The TROPICS team studies these questions to develop reliable and efficient reverse AD models and the code analysis that these models call for.

At the same time, we believe that AD cannot be a simple black-box tool used in Scientific Computing, except for relatively simple codes. In real applications, efficiency is achieved only through the conjunction of an efficient AD tool and of an efficient dedicated numerical algorithm that uses the AD gradients in a clever way. The TROPICS team develops innovative methodologies to solve and optimize numerical systems with the help of AD, and validates them on industrial-strength applications e.g. in Computational Fluid Dynamics (CFD).

Last but not least, validation of efficient reverse AD models and application of AD-enabled methodologies to industrial applications both require that we develop a full-blown AD tool. Just a demonstration tool working on a simplified mini-language is out of question. The TROPICS team develops an industrial-size AD tool "TAPENADE", into which we integrate our models and algorithms, and that runs on large codes in actual languages. TAPENADE is actually distributed with research or commercial licenses, and we value the growing number of outside users as a mark of quality of our models.

In the following section, we will give just the necessary background to understand our objectives of 2005, our results, and our proposed objectives for the next period.

2.2.1 A short description of AD problematics

Automatic or Algorithmic Differentiation (AD) differentiates *programs*. An AD tool takes as input a source computer program P that, given a vector argument $X \in \mathbb{R}^n$, computes some vector function $Y = F(X) \in \mathbb{R}^m$. The AD tool generates a new source program that, given the argument X, computes some derivatives of F. To this end, AD assumes that P represents all its possible run-time sequences of instructions, and it will in fact differentiate these sequences. Therefore, the *control* of P is put aside temporarily, and AD will simply reproduce this control into the differentiated program. In other words, P is differentiated only piecewise. Experience shows this is reasonable in most cases, and going further is still an open research problem (*see Objective 2, section 2.5*). Any sequence of instructions is then identified with a composition of vector functions. For a given control:

$$\begin{array}{lcl} P & \text{is} & \{I_1; I_2; \ldots I_p;\}, \\ F & = & f_p \circ f_{p-1} \circ \ldots \circ f_1, \end{array} \tag{1}$$

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

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

which can be mechanically translated back into a sequence of instructions I'_k, and these sequences inserted back into the control of P, yielding program P'. This can be generalized to higher level derivatives, Taylor series, etc.

In practice, the above Jacobian $F'(X)$ is often far too expensive to compute and store. Notice for instance that equation (2) repeatedly multiplies matrices, whose size is of the order of $m \times n$. Moreover, most problems are solved using only some projections of $F'(X)$. For example, one may need only *sensitivities*, which are $F'(X).\dot{X}$ for a given direction $\dot{X}$ in the input space. Using equation (2), sensitivity is

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

which is easily computed from right to left, interleaved with the original program instructions. This is the principle of the *tangent mode* of AD, which is the most straightforward, available in most AD tools.

However in optimization, data assimilation adjoint problems, or inverse problems, the appropriate derivative is the *gradient* $F'^*(X).\overline{Y}$. Using equation (2), the gradient is

$$F'^*(X).\overline{Y} = f'^*_1(X_0).f'^*_2(X_1). \ldots .f'^*_{p-1}(X_{p-2}).f'^*_p(X_{p-1}).\overline{Y}, \tag{4}$$

which is most efficiently computed from right to left, because matrix×vector products are so much cheaper than matrix×matrix products. This is the principle of the *reverse mode* of AD.

This turns out to make a very efficient program, at least theoretically. The computation time required for the gradient is only a small multiple of the run-time of P. It is independent from the number of parameters n. In contrast, notice that computing the same gradient with the *tangent mode* would require running the tangent differentiated program n times.

However, we observe that 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 $f'^*_{k+1}(X_k)$. This is the main problem of the reverse mode (*see Objective 1, section 2.4*). There are two popular strategies for addressing this problem:

- **Recompute All (RA):** the X_k is recomputed when needed, restarting P on input X_0 until instruction I_k. The TAF tool uses this strategy. Brute-force RA strategy has quadratic time cost with respect to the total number of run-time instructions p.
- **Store All (SA):** the X_k are restored from a stack when needed, *cf* figure 1. This stack is filled during a preliminary run of P, that additionally stores variables on the stack just before they are overwritten. The ADIFOR and TAPENADE tools use this strategy. Brute-force SA strategy has a linear memory cost with respect to p.

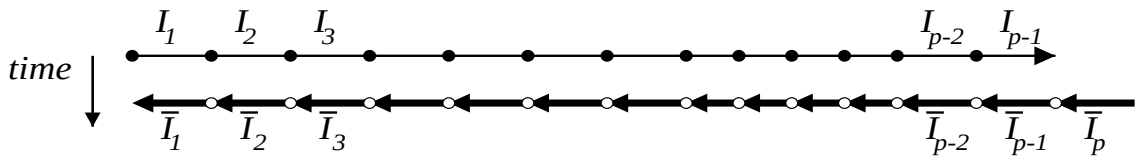

Figure 1: The "Store-All" tactic

On industrial-size applications, the time cost of brute-force RA is unacceptable, and so is the memory cost of brute-force SA. Both RA and SA strategies need a special storage/recomputation trade-off. Incidentally, they become very similar then. This trade-off is called *checkpointing*. Since TAPENADE uses the SA strategy, let us describe checkpointing in this context only. The plain SA strategy (*cf* figure 1) applied to instructions I_1 to

I_p builds a differentiated program where an initial "forward sweep" runs the original program and stores intermediate values (black dots), and is followed by a "backward sweep" that computes the derivatives in the reverse order, using the stored values when necessary (white dots). Checkpointing a fragment **C** of the program is illustrated on figure 2. During the forward sweep, no value is stored while in **C**. Later, when the backward sweep needs values from **C**, the fragment is run again, this time with storage. One can see that the maximum storage space is grossly divided by 2. This also requires some extra memorization, called a "snapshot", to restore the initial context of **C**. This snapshot is shown on figure 2 by slightly bigger black and white dots. Checkpoints can be nested. In that case, a

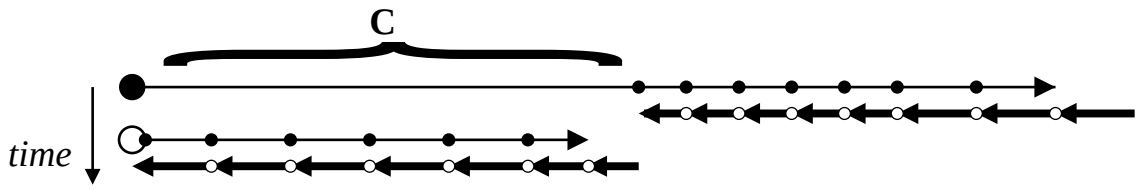

Figure 2: Checkpointing **C** with the "Store-All" tactic

clever choice of checkpoints can make both the memory size and the extra recomputations grow only like the logarithm of the size of the program. However, the optimal choice is in general NP-hard to find.

2.2.2 Applications of AD to Scientific Computing

Scientific Computing provides a lot of applications for derivatives, the most popular being Simulation, Optimization and Inverse Problems. Even in stochastic approaches to optimization such as evolutionary methods, derivatives are used increasingly.

There are several approaches to obtain these derivatives, and we must contrast the AD approach with the others. Let us concentrate on obtaining a gradient. At one extreme, one can write an *adjoint system*, then discretize it and program it by hand. The adjoint system is a new system, deduced from the original equations, and whose solution, the *adjoint state*, leads to the gradient. A hand-written adjoint is very sound mathematically, because the process starts back from the original equations, but this involves a long additional implementation phase. One can use mathematical knowledge of the problem to write remarkably efficient code. Strangely enough, the separate discretization for the original and for the adjoint code causes subtle problems when optimizing with descent directions: It computes a discrete gradient with is not exactly the gradient of the discrete functional.

Therefore it is safer to compute the adjoint of the discrete functional itself, i.e. apply the reverse mode of AD to the program. Although this has often been done in the past, we do not advocate writing the reverse differentiated code by hand, i.e. playing the role of the AD tool. Of course a human programmer can always be more clever than a software tool, but AD tools have made tremendous progress and will continue steadily. More importantly, it is highly desirable that the original simulation code evolves, and rewriting the adjoint code by hand every time is impractical.

Even if users now understand the interest of *Automatic* Differentiation, they are often not satisfied with the reverse-differentiated code. Reverse AD often means massive use of storage and recomputation, requiring post-AD hand transformation to recover efficiency.

See [1],[2] for a catalog of post-AD manipulations. The TROPICS team aims at progressively improving the AD tools, and TAPENADE in particular, to replace these post-AD manipulations by automated AD strategies, possibly triggered by directives in the original code.

Now we will give a quick panorama of the applications of AD that we know of, for Simulation, Optimization and Inverse Problems. Simulation of complex systems can be locally replaced by reduced models, i.e. local linearizations of the system around the current configuration. For a better approximation, the reduced model can use second-order derivatives. One can also approximate the local behavior of a simulation in a stochastic way, using the so-called method of moments, which also involves first- and second-order derivatives. Actually, second derivatives appear at several places in this panorama, which motivates our work on that topic (*see Objective 3, section 2.6*). The accuracy of a simulation also depends on the quality of the discretization. AD gradients can be used for mesh adaptation (*see Objective 4, section 2.7*). Optimization is maybe the most natural application of AD gradients. Optimization is one degree more complex than plain Simulation, and efficiency of AD gradients becomes crucial. This motivates algorithmic improvements inside the reverse AD model (*see Objective 1, section 2.4*), and specialized optimization algorithms (*see Objective 4, section 2.7*). In the case of *Robust Optimization*, stability of the optimum with respect to optimal parameters is required. This introduces second derivatives in the optimization problem. 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 are the altitude and velocity at the surface. Another example is *data assimilation* in weather forecasting[3]. In general, the problem is stated as a least square problem, between the observed values and the simulated values. This rapidly boils down to solving an adjoint problem, which can be done through AD. Here also, second derivatives are required more frequently, in particular to study correlations and sensitivity of the estimated parameters.

In the special case of steady state simulations, many works have shown that the adjoint can be computed far more efficiently than by reverse AD of the complete pseudo-time advancing scheme at a tremendous memory cost. For instance in the "piggy-back" approach[4], computation of the adjoint state uses the iterated states in the direct order. Alternatively, most researchers use only the fully converged state to compute the adjoint. This is usually implemented by a delicate and error-prone modification of the code generated by AD. The TROPICS team is proposing a methodology that combines AD and hand coding [10, 17]. This methodology can be extended to second derivatives (*see Objective 3, section 2.6*).

2.3 Objectives for the evaluation period

To begin with, here is an exact copy of the objectives we had given at the beginning of the present evaluation period, in march 2005.

——— < *Start 2005 objectives copy* > ———

[1] P. Hovland, B. Mohammadi, C. Bischof, "Automatic Differentiation of Navier-Stokes computations", *research report number MCS-P687-0997*, Argonne National Laboratory, 1997.

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

[3] F. LeDimet, O. Talagrand, "Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects", *Tellus 38A*, 1986, p. 97–110.

[4] A. Griewank, C. Faure, "Reduced Gradients and Hessians from Fixed Point Iteration for State Equations", *Numerical Algorithms 30(2)*, 2002, p. 113–139.

Computer Science aspects: improve AD models. Our objectives are:

- **(o1) Merge Store-All and Recompute-All:** one open problem is to merge apparently conflicting strategies for the reverse mode. We believe that a data-dependence approach can capture into a single framework the Store-All and the Recompute-All strategies of the reverse mode, as well as Static Single Assignment and the interaction with Checkpointing. We will address this problem soon.
- **(o2) Completely formalize intermediate values storage and checkpointing:** In reverse-mode AD, many values are stored either because they will be used in the partial derivatives in the reverse sweep or because they are in a checkpoint. Such storage is often redundant because there is no common formalization of these mechanisms. We can define a precise model of reverse differentiation, and from it define the minimal set of variables that must be stored at each point.
- **(o3) Make checkpoints more flexible:** Checkpointing is absolutely unavoidable on large codes, for example for optimization of unsteady processes. Checkpointing arbitrary pieces of code, designated by the end-user, is highly desirable. Successive checkpoints must share their stored variables whenever possible, and this requires tradeoffs. There exist other flavors of checkpointing (*"reverse checkpoints"*) that we didn't investigate yet, that would deserve deeper studies.
- **(o4) Analyze Discontinuities:** AD of programs around discontinuities is still an open problem. The problem cannot be simply "solved" in the current state of the art. During Mauricio Araya-Polo's PhD work, we will study a special differentiation mode that will evaluate the size of the neighborhood of the current input values in which no discontinuities occur.

About tool development, we know that TAPENADE can be largely improved, although it now surpasses the functions of its predecessor ODYSSÉE. TAPENADE still misses treatment of user directives for AD, and a decent analysis of pointers and dynamic memory. These problems will be even more important when C becomes an input language (started in 2005). Pointers and memory allocation are still a challenge for the other AD tools, as well as differentiation of C. Nevertheless, the biggest challenge for AD tools is definitely differentiation of object-oriented languages (e.g. C++), which are spreading more and more in Scientific Computing. To address this, we need to collaborate more with specialists of static analysis and compilation of object languages.

Numerical Science aspects: now that AD provides analytic derivatives, convince numerical scientists to use those. The open problem here is to design new efficient ways to use AD adjoints in optimization programs. We have promoted an approach for optimization of a steady-state simulation using AD first-order gradients. Our objectives are:

- **(o5) Provide and use higher-order derivatives,** and in particular Hessians, which are of course useful in optimization. Combinations of the existing first-order differentiation modes are a possible answer, but efficiency requires specific modes.
- **(o6) Optimize unsteady simulations,** which are one degree of complexity higher than steady simulations, and require powerful checkpointing schemes.
- **(o7) Continue coupled shape-mesh optimization.** A thesis is starting to complete our contribution on this topic.

——— < *end 2005 objectives copy* > ———

The recommended format for the sequel is one section per objective, and suggests an average of 4 objectives. This is perfectly reasonable. We will therefore group our objectives of 2005 in the following 4 big objectives. Our objectives in 2005 also clearly mentioned further development of our AD tool TAPENADE. Although the format of this report exiles Software Development towards the later "Software" section 3.1, we view development of TAPENADE as an essential objective of the team.

2.4 Objective 1: Use Data-Flow analysis to improve reverse AD (o1, o2, and o3)

We address the specific problems of the reverse mode of AD. We use Data-Flow analysis to reduce the memory and runtime penalties caused by data-flow reversal.

2.4.1 Personnel

Laurent Hascoët, Valérie Pascual, Mauricio Araya-Polo, Benjamin Dauvergne, and external collaboration with Uwe Naumann (RWTH Aachen), Jean Utke (Argonne Nat. Lab.), Mike Fagan (Rice Univ.).

2.4.2 Project-team positioning

From one point of view, this problem is interesting only to the small community of researchers working on reverse AD, plus all end-users. In fact, mostly teams who develop a software tool for reverse AD are working on this topic. This leaves us with

- the ADOL-C team in Dresden, Germany,
- the developers of the AD extension of the NAG F90 compiler in Hatfield, UK and Aachen Germany,
- and the developers of the OpenAD platform in Argonne, IL, USA.

Because it relies on overloading, ADOL-C reverse mode stores more information than TAPENADE's reverse mode. In particular they also store a part of the original operations in addition to the intermediate values. Their research effort is more focused on reducing this "operations stack".

Both the AD-enabled NAG F90 compiler and OpenAD try to optimize the reverse mode by reducing the operations count on the computation graph of the derivatives, for any given basic block of instructions. Although this is a very interesting theoretical problem (recently proved NP-hard by Uwe Naumann), we believe it is not the most needed improvement for a practical AD tool. OpenAD features a number of static data-flow analysis, some of them needed to build good checkpointing strategies in the reverse mode. But to our knowledge, OpenAD does not optimize the memory storage of reverse differentiated code. Nevertheless they are studying these questions and we have a fruitful collaboration with them and Uwe Naumann on this topic.

The private company FastOpt, developing the AD tool TAF, are certainly very close to this topic. TAF has original strategies that would be worth studying, but as a private company, they communicate very little on their AD models and algorithms.

From another point of view, this topic is just one of the many program analysis and code optimization problems. The tools we use (data-flow, data-dependence graphs, use-def or def-use chains ...) are well established and described in textbooks on parallelization

or compilation [5]. We have occasional contacts with teams doing program analysis, such as ALCHEMY (INRIA Orsay), and ARENAIRE (LIP Lyon).

2.4.3 Scientific achievements

By nature, the derivative computations in reverse mode require intermediate values from the original program, in the inverse of their normal production order. The overhead of reverse AD thus comes from restoration (through storage or recomputation) of values required in the derivatives. Naïve implementation causes restoration that is often useless, redundant, or inefficient. In particular, one issue is the coupling between the main restoration strategies. This results in many complaints of AD users arguing with reason that they need to edit the differentiated code by hand to make it efficient. This also results in a growing basis of "oral tradition" rules to go around these weaknesses of restoration. Generally speaking, we advocate a more rigorous approach, based on a precise data-flow model of restoration. In the long term, we aim at providing a uniform description of all restoration strategies that encompasses the existing and exhibits optimal compromises. Several of our improved data-flow-based algorithms, now implemented in TAPENADE, actually reduce the need for post-differentiation hand editing.

We distinguish for convenience two nested granularity levels for the restoration problem:

- At the fine-grain level of sequences of individual instructions, the question is which intermediate values must be restored, and through which combination of storage, recomputation and inversion.
- At the coarse-grain level of nested checkpoints, one question is to find the smallest snapshots, taking into account the coupling between checkpoints. When this is clarified, a more ambitious question is to improve the very location of the checkpoints.

We build on our past study of the data-flow equations of the static analysis involved in reverse AD [6] that clarified the fine-grain level, without checkpoints. From this, the set of intermediate values required is well established.

During this period, we continued the fine-grain study by taking into account a new restoration strategy: inversion. A trivial example is to inverse increments of array indices e.g. `j = j+2` by inserting `j = j-2` at a correct place in the adjoint code, but there is more to it than plain detection of induction variables. This work was done in collaboration with Jean Utke and Uwe Naumann [16]. In a first step, we restrict to the important subcase of control values i.e. array indices, pointer addresses, test results, as opposed to floating point values in computations. We propose a conceptual framework akin to a data-dependence graph that captures three restoration strategies (inversion, recomputation, and storage), along with a heuristic to find a good combination of those.

At the fine-grain level, another performance issue is data locality in the differentiated code. Keeping the variables distinct from their derivatives ("association by name") preserves the structure of the original variables, which is crucial e.g. to I-O. On the other hand, keeping the derivative attached to the original variable ("association by address") improves data locality in differentiated instructions. In a joint work with Jean Utke and Mike Fagan [28], we present this question and explore practical solutions that in effect represent hybrids of the two approaches.

[5] S. MUCHNICK, "Advanced compiler design and implementation", *Academic Press*, 1997.

[6] L. HASCOËT, M. ARAYA-POLO, "The adjoint Data-Flow analyses: formalization, properties, and applications", *Proceedings the AD 2004 conference*, Chicago, 2004.

At the coarse-grain level and during this period, we first analyzed the data-flow of checkpointing, yielding a precise characterization of all possible memory-optimal options for snapshots [27]. This characterization is formally derived from the structure of nested checkpoints and from classical data-flow equations. It captures forms of coupling where reducing one snapshot may lead to a higher number of values being stored at the fine-grain level. We experimented several of these optimal options, using TAPENADE, on a number of real codes. Although no option is uniformly better, we observed that the so-called "lazy snapshot" option gives best performances in general. This option is now the default strategy in TAPENADE, yielding a significant gain (around 10%) over the previous more naïve checkpoints.

Still at the coarse-grain level, we studied the question of optimal placement of checkpoints in a large application. This question has a proved optimal solution only in the case of a fixed-length time-stepping loop with no checkpointing inside each time-step. In contrast, we considered the case of an arbitrary program call tree, with the option to place a checkpoint or not at each subroutine call site. Using our data-flow equations for snapshots, and either some static analysis or some run-time profiling, we can evaluate the performance of each placement of checkpoints, and look for an optimal. Benjamin Dauvergne has worked on this question with Uwe Naumann in Aachen. The optimal placement problem is NP-hard [7]. Benjamin Dauvergne then developed heuristics to find good enough placement of checkpoints [15] without running an exhaustive (exponential) search. He obtained promising improvements on two large applications. Incidentally, this required the development in TAPENADE of the `NOCHECKPOINT` directive, now available for all users.

During the evaluation period, we also started to study reverse AD of message-passing parallel programs, from the data-flow point of view. This preliminary joint work [35, 37] with Jean Utke and Uwe Naumann is certainly one of the most promising perspectives of future research. More at an implementation level, we have fully automated the "`II-LOOP`" checkpointing strategy in TAPENADE. This strategy applies a very efficient specialized checkpointing scheme to loops with Independent Iterations, i.e. a sub-class of parallel loops. It can now be triggered by the end-user with a `II-LOOP` directive in the source program.

2.4.4 Collaborations

• Joint work and paper [28] with Mike Fagan (Rice) and Jean Utke (Argonne) on alternative representations for derivatives variable and data structures.
• Joint work (4 weeks meeting) and paper [16] with Jean Utke and Uwe Naumann (Aachen) on fine-grain restoration of control-flow information.
• Joint work with Uwe Naumann on the optimal placement of checkpoints. Benjamin Dauvergne spent 4 weeks in Aachen with Naumann's team.
• Ongoing work with Jean Utke and Uwe Naumann on reverse differentiation of MPI parallel communication primitives. Joint articles [35, 37].

2.4.5 External support

2.4.6 Self assessment

This objective is central for the TROPICS team. We are convinced that inverse problems and optimization problems will spread even further in Scientific Computing, and that

[7] U. NAUMANN, "Call Tree Reversal is NP-Complete", *Advances in Automatic Differentiation, Springer*, 2008.

gradient-based methods are the main key to them. We acknowledge gradients are not the only derivatives of interest, but our choice is to invest most of our work on the reverse mode of AD, and to apply our program data-flow analysis background on reverse AD. We believe our data-flow approach has brought something to the AD community.

One achievement of this period is the complete data-flow formalization of snapshots and of their coupling with other improvement strategies in reverse AD. This gives us strong confidence in the correctness and efficiency of the checkpointing mechanism implemented in TAPENADE.

We are also pleased with the data-flow formalization at the fine-grain level for reversal of address computation, although this needs further work before it can be used inside an AD tool.

Results on optimal placement of checkpoints on the call tree are interesting from a theoretical viewpoint, but we need to spend the development effort to turn them into a practical component of our AD tool.

We are extremely satisfied with the ongoing collaboration on these topics with the Argonne and Aachen teams. Clearly we develop competitor AD tools, but we are building a common view of their internal algorithms. This is particularly important for a small research team such as TROPICS.

2.5 Objective 2: Analyze discontinuities (o4)

We want to give some answers to one weakness of AD compared to inefficient but sturdy approaches such as Divided Differences: AD can silently return wrong derivatives in presence of discontinuities of the computed function.

2.5.1 Personnel

Mauricio Araya-Polo, Laurent Hascoët

2.5.2 Project-team positioning

Although the question of non-differentiability is generally recognized as a thorny problem that can ruin hopes of gradient-based optimization, there are very few works addressing the question. Most of these works stay at a very abstract level. In the domain of interval arithmetic, Baker Kearfott [8] proposes and implements an extension to approximate intervals in the case of non-smooth constructs. In the differentiation domain, some authors proposed to introduce sub-differentials (i.e. an interval around the undefined differential), but they generally remain at a theoretical level due to the incurred computational complexity.

Griewank and Walther devote the whole section 14 of their classical AD book [9] to this question. They identify the problem in the same manner, but they concentrate on a mathematical formalization of the problem, classifying the kinds of non-differentiability that may occur. They address the problem of giving one-sided derivatives at the places where the derivative is undefined. In particular they advocate use of Laurent series, which may have a reasonable cost when one restricts to one user-given direction in the input space. This is interesting because we ended up making the same restriction: Apparently,

[8] R. Baker Kearfott, "Interval extensions of non-smooth functions for global optimization and nonlinear systems solvers", Computing, 1996

[9] A. Griewank, A. Walther, "Evaluating Derivatives: principles and techniques of Algorithmic Differentiation", *2nd edition*, SIAM, 2009

this confirms that our initial objective of finding the complete differentiable domain in one single run is out of reach.

2.5.3 Scientific achievements

Automatic Differentiation gives analytical derivatives, e.g. gradients, based on local properties of the function at the given input point. These derivatives are then used to extrapolate the behavior of the function around this point. Since the analysis is local, these derivatives do not take into account discontinuities that may occur in the neighborhood of the current point. These discontinuities basically correspond to conditionals in the source program.

In a source program, the sources of non-differentiability are the control switches, plus the language intrinsics that may contain control switches. These places are easily located. If a change of the input point can make one of these switches swap at run-time, then there is an input point for which the computed function is a priori non-differentiable.

For each run-time switch, involving some intermediate variables, we can find for which values of the intermediate variables the non-differentiability occurs, at least at first order. This gives us a half-space of the space of intermediate variables, containing the current values. As far as this switch is concerned, the computed function remains differentiable in this half space.

The difficulty is to combine all these half-spaces into one subdomain of the space of inputs, in which the computed function is differentiable. We explored several strategies and evaluated their complexity. Reverse AD allows us to propagate *one* half-space on intermediate variables into one half-space on the inputs. But this should be done once for each run-time switch, and the cost is unreasonable. We had to abandon this reverse-based strategy.

Therefore, Mauricio Araya-Polo proposed a strategy to evaluate the distance to the nearest discontinuity along one given direction in the input space. This is done by extension of the tangent mode of AD. During execution of the tangent differentiated program, each encountered switch accumulates its constraints into one single validity interval around the input point, along the given direction.

We proposed two possible uses of this extension:

- One can compute a polyhedral approximation of the differentiable domain around the current input point, by repeatedly running the extended tangent code for several directions in the input space. For instance, if these directions are set to the Cartesian basis of the input space, the differentiable domain can be computed at a cost proportional to the number of inputs.

- Inside an optimization process, one uses the reverse mode of AD to compute a gradient and from it choose a descent direction. At this moment, one can run the cheap extended tangent code for this descent direction. This will tell to which extent one can follow the descent direction safely.

Mauricio Araya-Polo made several demonstrative experiments on real applications. He published partial results in a journal article [5] and a complete discussion in his PhD thesis [1]. The distributed version of TAPENADE now has a command-line option to trigger this functionality. Several uses were reported by external users, although on small applications or for educational purposes.

2.5.4 Collaborations

2.5.5 External support

2.5.6 Self assessment

This problem of non-differentiability is known as very disturbing to every AD researcher. To our knowledge, this is the first attempt at an effective answer that can be implemented, and we did implement it.

However, it is true that this question is consistently overlooked in AD of real applications. Most programs are full of switches. Still, people happily apply AD to then to obtain gradients and it works! One possible reason of this surprising robustness is that industrial-quality optimization routines always perform a line search along the descent direction, so that going through a non-differentiability will not make optimization fail but rather make it a little slower.

Our feeling is that this work both has a theoretical interest and provides end-users with a useful quantitative tool. But until some end-user application clearly shows a need for a better estimation of the differentiability domain, we do not plan to invest further work into this topic.

2.6 Objective 3: Provide and use higher-order derivatives (o5)

We study the question of Higher-Order derivatives, beginning with second derivatives. This work goes along two directions. One is to study sophisticated uses of the Hessian matrix in Scientific Computing. The other is to use and improve AD to effectively compute this Hessian. This objective corresponds to the PhD work of Massimiliano Martinelli [2].

2.6.1 Personnel

Massimiliano Martinelli, Alain Dervieux, Laurent Hascoët

2.6.2 Project-team positioning

The question of efficient computation of second derivatives in CFD was studied in the pioneering works of Taylor-III et al [10]. For our part, we started this study in close collaboration with Mike Giles team in Oxford.

We are aware of works that try to take advantage of the sparsity of the Hessian matrix, described in detail in Andreas Griewank's book. In our work however, we only took advantage of the Hessian's symmetry. Actually, the Hessian in our CFD application is not very sparse.

Other INRIA teams concentrate on novel uses of Hessians in optimization, rather than on the ways of obtaining these Hessians. This is the case for the OPALE team, in which Massimiliano Martinelli has spent year 2008 as a Post-Doc.

2.6.3 Scientific achievements

During this period, we concentrated on second derivatives e.g. Hessian matrices. Although some authors advocate the use of third-order derivative tensors, we left this aside. Also, we are more interested in complete derivative objects such as rows of the Hessian, than in directional Taylor expansions. In fact, we believe Taylor expansions are more easily computed with operator-overloading AD tools, such as ADOL-C.

[10] Taylor III, A.C., Green, L.L., Newman, P.A., Putko, M.M. "Some advanced concepts in discrete aerodynamic sensitivity analysis", *AIAA Journal 41(7)*, 2003

Like we did in the past for first derivatives, we considered the case of steady-state solvers. This very frequent case allows us to develop sophisticated strategies to avoid reverse differentiation of each pseudo-time step, because the intermediate states in the pseudo time-stepping have no physical meaning. This is what we did in the past for first derivatives, and we did it this time for second derivatives [31].

Starting back from the mathematical equations of a constrained functional computed through a steady-state iterative solver, we actually found two possible strategies for computing the second derivatives of the functional. These strategies both take advantage of the steady-state nature to compute only a limited amount of expensive derivatives. One strategy eventually makes use of Hessian-times-vector products obtained by repeated Tangent-on-Tangent differentiation. The other obtains these products by Tangent-on-Reverse differentiation.

A careful cost analysis of the two strategies, with respect to the number of input parameters n shows that the Tangent-on-Reverse approach outperforms Tangent-on-Tangent only above a relatively high n. However this contradicts former works that claimed that Tangent-on-Tangent is always faster. This comes from a better accuracy in our cost analysis. We validated this cost analysis on a couple of examples [34].

Massimiliano Martinelli applied these results to several domains of Scientific Computing where second-order derivatives are needed. One particular application is the estimation of uncertainties [2] on very complex systems with a computer-intensive high fidelity Navier-Stokes model. It is too expensive to search the statistical properties of the system with Monte-Carlo methods on the simulation itself, so we do it on a reduced-order model that is built from first and second order derivatives obtained by AD [31, 34]. Another application is robust design, where some sensitivity terms, obtained by AD, are themselves included into the cost functional. This naturally introduces second-order derivative terms in the optimization process [19].

Repeated application of Automatic Differentiation raises new questions in the case of Tangent-on-Reverse. The restoration mechanism incurred by reverse AD, at least with the Restore-All strategy of TAPENADE, uses a stack that is accessed with external `PUSH` and `POP` routines. In the following tangent differentiation step, the derivatives of these external routines must be provided by the end-user. We realized how easy it is to write these derivatives wrong, and even when they are written correctly, the derivative code proves very inefficient. Actually what we need is a correspondence between each matching `PUSH` and `POP`, expressed e.g. with directives. These directives must be set by the first, reverse differentiation, and used by the second, tangent differentiation [34]. Implementation of this extension is going on in TAPENADE.

2.6.4 Collaborations

- We had very profitable discussions with Mike Giles (Oxford), who uses TAPENADE in a project with Rolls-Royce. They use second derivatives in the design of turbines.
- We collaborate with team OPALE on the European project NODESIM-CFD, about reduced models and robust design.

2.6.5 External support

Martinelli's work was supported by the NODESIM-CFD "STREP" European project.

2.6.6 Self assessment

Repeated application of AD is a classical way to obtain second derivatives, and we know TAPENADE has already been used for this in the past. Still, this raised several specific problems for the Tangent-on-Reverse approach. This work was the right occasion to experiment with this approach in-house, and to propose solutions. Actual implementation of these solutions in TAPENADE is slightly lagging behind.

Also, it is important that we now have a clear comparison between Tangent-on-Reverse and Tangent-on-Tangent. Actually, an open question now is "what about Reverse-on-Tangent?".

Putting up a sophisticated strategy to compute Hessian elements for steady-state computation is a key to the application of AD to robust design. We are also beginning to see applications in sensitivity studies in Earth Sciences. Some applications are showing up where the given simulation is now *unsteady*. Our sophisticated strategy above does not apply any more. Further study is necessary there and, until this is done, the Tangent-on-Tangent approach is probably the safest choice.

We do not think it is worth going up to third-order derivatives with the repeated application approach. There is always an overhead, and program analysis becomes increasingly costly. For really high order derivatives, we believe that the right approach is operator-overloading AD computing directional Taylor expansions.

We may continue some research on Hessians in the future, but at the moment we do not plan to invest more in this direction.

2.7 Objective 4: Study applications to optimization (o6 and o7)

Automatic Differentiation, especially in reverse mode, is rarely a black-box tool that scientists can simply use in their numerical computation. Very frequently, there must be a common work to develop sophisticated numerical algorithms that use AD gradients in a more efficient way. Therefore, the TROPICS team not only develop improvements to the AD model, described in the previous sections, but also study new algorithms for key questions in scientific computing. The targets we have chosen are optimal control, shape optimization, and control of approximation errors by mesh optimization.

2.7.1 Personnel

Alain Dervieux, Laurent Hascoët, Massimiliano Martinelli, Anca Belme, Benjamin Dauvergne, and collaboration with Youssef Mesri (INRIA team SMASH), Fréderic Alauzet, and Adrien Loseille (INRIA team GAMMA).

2.7.2 Project-team positioning

Research teams on Optimization are far more numerous than on Automatic Differentiation. Incidentally, many of them do not consider gradient-based optimization but rather stochastic approaches. This sightly reduces the set of our potential end-users.

Rather naturally, we chose to collaborate with INRIA teams OPALE, SMASH (in Sophia-Antipolis) and GAMMA (in Rocquencourt).

2.7.3 Scientific achievements

We have been working mostly on two classes of applications, optimization, optimal control on one hand, and on the other hand control and correction of approximation errors due to mesh discretization.

One present frontier in industry research groups is optimization. Simulation itself is well mastered. Optimization is hard because the number of optimization parameters is high, typically several hundreds, particularly in CFD optimal shape design. Alain Dervieux and Francois Beux edited a collection of papers on the problematic of shape design [13]. The reverse mode of AD is absolutely necessary in this context, because it is the most efficient way to get the gradients when the number of parameters is large. However, a plain reverse differentiation of the complete simulation algorithm can be unnecessarily costly, especially for steady-state simulations.

During this period, we have further developed our general strategy to build the adjoint of a steady-state simulation, using a combination of reverse AD and sophisticated hand-written code. The central idea is to apply reverse AD on the discrete state residual, which gives (but implicitly) the matrix that appears in the adjoint state equation. Therefore, we resort to a matrix-free solver (GMRES) to get the adjoint state. We speed up convergence by preconditioning with the first-order (in space) Jacobian of the state equation, combined with an incomplete factorization ILU(k). In particular neither the pseudo-time stepping nor the linear solver are reverse-differentiated [22]. We gave particular attention to the issues of efficient computation of a large-scale adjoint system [11], efficient preconditioners for this computation [7], and efficient optimization algorithms [10]. The same philosophy was applied by Massimiliano Martinelli for second derivatives [2].

We addressed the problem of approximation errors due to mesh discretization in collaboration with INRIA teams GAMMA and SMASH. We managed to transform the mesh adaptation problem into a differentiable optimal control problem, on which we can apply the same strategy as for CFD optimization. To this end, we introduced a new methodology that consists in stating the mesh adaptation problem in a purely functional form: the mesh is reduced to a continuous property of the computational domain, the "continuous metric", and we minimize a continuous model of the error resulting from that metric. Then the problem of searching an adapted mesh is transformed into the search of an optimal metric. During the thesis of Youssef Mesri, the problem of coupling mesh adaptation and shape optimization has been addressed by the design of new algorithm, the fixed point adaptation-optimization algorithm [3].

In the case of mesh interpolation minimization, the optimum is given by a closed formula and gives access to a complete theory demonstrating that second order accuracy can be obtained on discontinuous field approximation [8]. In the case of adaptation for Partial Differential Equations such as the Euler model, we need an adjoint state that we obtain by AD. We end up with a minimization problem for the metric which in turn is solved analytically [4, 12, 14, 17].

We started very recently to consider the problem of the correction of approximation errors. This new subject is addressed jointly by teams OPALE and TROPICS. It is bound to become an important application of AD. We investigate the two types of correctors, by direct linearizations and Defect Correction, or by the adjoint-based functional correction. The purpose is to apply these methods to large unsteady flow simulations. These studies will contribute to the approximation error section of project NODESIM-CFD.

The work above is focused on the case of steady-state simulations. We also studied several unsteady applications. The strategies described above do not apply to unsteady simulations, because each time-step is physically meaningful and is taken into account for the global cost functional. Therefore we essentially apply reverse AD to the complete simulation, and implement strategies directly into the AD tool, triggered by user directives, to produce a more efficient code. The Numerical Science aspects are therefore less important here, but the Computer Science aspects and tool implementation play a larger role. During the evaluation period we have successfully built the adjoint code of the following

large unsteady applications, each time in collaboration with the end-user.

- With INRA (agronomy) we differentiated STICS and SAIL, two Fortran codes of 26 000 lines [25, 18].
- With IFREMER, LOCEAN lab of Paris VI university, and IMAG Grenoble, we differentiated the OPA ocean circulation model, in the small "configuration" GYRE and the large configuration NEMO. This is a Fortran90 code of 120 000 lines [29, 39].
- With ANDRA (geology) and the JAD lab. of university of Nice, we differentiated the TRACES code, a Fortran90 code of 22 000 lines.

Comments apply to all these codes: first, we acknowledge that reverse AD never worked at first try on these codes. There was always a couple of month's step of debugging TAPENADE, to catch up with the recent features in recent codes. For instance the TRACES code was our first application with intensive pointer allocation and manipulation. Then comes efficiency questions. The adjoint codes benefited a lot from the TAPENADE directives `II-LOOP` for parallel loops, from the refined computation of snapshots that we have developed, and from the `NOCHECKPOINT` directive to control checkpointing. This is discussed in [15].

2.7.4 Collaborations

- Collaboration with team SMASH on the European project HISAC: co-advising of Youssef Mesri.
- Collaboration with team GAMMA on the European project HISAC: collaboration with Fréderic Alauzet and co-advising of Adrien Loseille.

2.7.5 External support

- The "HISAC" European IP project provided partial support for Adrien Loseille and Youssef Mesri
- The "LEFE" French ANR project provided partial support to advertise our results on the adjoint of OPA NEMO code.

2.7.6 Self assessment

Concerning the optimal design (with optimal control methods) relying on steady numerical models, we think that a cycle is more or less completed:

- we have identified the necessary extensions of AD methods, such as `II-LOOP`s, we have worked out the new methods or new extensions, introduced them in TAPENADE, experimented with our CFD platforms and with CFD software from partners, diffused them so that they are applied by other teams, in particular in two European projects (common paper with Dassault).
- we have identified and developed complementary numerical methods, trying to answer various questions, arising in particular from the three European projects, AEROSHAPE (finished before the evaluation period), HISAC, and NODESIM-CFD: efficiency of adjoint based design with one-shot methods and multilevel preconditioning on shape iteration, combination of mesh adaptation and shape optimization in a convergent loop.

The above aspects are complementary to the questions of stochastic and robust optimization, and of reduced models addressed by other INRIA teams (OPALE) or outside (IMF-Toulouse) with our without our contribution. We have hesitated and decided not to develop a parallel demonstration platform for shape design. This was for two main reasons, first it is a large investment just for "demonstrating", second, we planned to address parallel Automatic Differentiation a little later, i.e. in the coming period.

The case of optimization in unsteady turbulent model is a difficult question which we may address in the future. But before that, we are still interested in other applications of linearizations and adjoints in steady and unsteady flows, involving error correctors and mesh adaptation.

Concerning mesh adaptation, the methods built with GAMMA, are the first ones combining in a natural way adjoint and Hessians. Mesh adaptation studies will be pushed further in cooperation with the GAMMA team and Lemma company. This will be a large part of our efforts. Our plan involves the extension of our methods to many steady (compressible Navier-Stokes) and unsteady (free interfaces, LES turbulence,...) CFD models.

Concerning unsteady flows, we plan to re-design the implicit time advancing in order to keep it sophisticated and even more efficient for hybrid RANS-LES, but also to have a global code easy to differentiate with an AD tool. The novel platform will be differentiated first for building strategies of numerical error correctors and static mesh adaptation (what is the best mesh for a complete time interval).

3 Knowledge dissemination

3.0.7 Publications

	2005	2006	2007	2008-9
PhD Thesis		1	2	1
H.D.R (*)				
Journal	1	5	1	7
Conference proceedings (**)	6	4	3	4
Book chapter				1
Book (written)				
Book (edited)				1
Patent				
Technical report		1	1	
Deliverable	2	2	2	2

(*) HDR Habilitation à diriger des Recherches
(**) Conference with a program committee

We are asked to mention here the major journals "in the field", then the papers by the team accepted in these journals in the evaluation period. Then same thing for conferences. The TROPICS team is working in two fields. The AD field uses conferences more than journals. The numerical field uses journals more than conferences. The adjective "major" is also very delicate. Some very generalist journals and conferences may be considered major (and indeed very selective), but a paper there hardly reaches its audience. Some focused journals and conferences are indeed very popular in their respective fields and have a better impact on the people we work with.

Journals:

1. Journal of Computational Physics: no paper

2. International Journal of CFD : [7]
3. Applied Numerical Mathematics: [8]
4. European Journal of Computational Mechanics: [6, 12, 19, 15]
5. Scientific Programming: [16]

Conferences:

1. AD2004 and AD2008 conferences, proceedings in Springer LNCSE: [23, 26, 34, 36]
2. International Conference on Computational Science (ICCS): [27]
3. ICFD Conference on Numerical Methods for Fluid Dynamics: [33]
4. ECCOMAS conferences: [30]

3.1 Software

The TROPICS team develops the Automatic Differentiation tool TAPENADE. TAPENADE progressively implements the results of our research about models and static analysis for AD. From this standpoint, TAPENADE is a research tool. Our objective is also to promote the use of AD in the scientific computation world, including the industry. Therefore the team constantly improves TAPENADE to meet the demands of our industrial users. From this standpoint, TAPENADE is also an industrial tool.

TAPENADE supports three modes of differentiation: tangent, vector tangent, and reverse. Although perfectly feasible, there is no vector reverse yet because we found no application for it. TAPENADE differentiates programs written in Fortran, up to the 95 standard, or in ANSI C. Like any program transformation tool, TAPENADE needs sophisticated static analysis in order to produce an efficient output. All these analysis are context-sensitive, flow-sensitive, and implement a limited form of array index analysis for more accurate results. The following analysis are a must for an AD tool and TAPENADE now performs them all:

- **Pointer (or Alias) analysis:** For any static program transformation, and in particular differentiation, it is essential to have a precise knowledge of the possible destinations of each pointer at each code line ("points-to analysis"). Otherwise one must make conservative assumptions leading to less efficient code.
- **Activity:** The end-user has the opportunity to specify which of the output variables must be differentiated (called the dependent variables), and with respect to which of the input variables (called the independent variables). Activity analysis propagates the independent forward and the dependent backward, in order to detect all "active" intermediate variables that both depend on the independent and influence the dependent. Only the active variables need to be differentiated, thus making the differentiated program smaller and faster.
- **Adjoint Liveness and Read-Write:** Programs produced by the reverse mode of AD show a very particular structure, due to their mechanism to restore intermediate values of the original program in the *reverse* order. This has deep consequences on the liveness and Read-Write status of variables, that we can exploit to take away unnecessary instructions and memory usage from the reverse differentiated program. This makes the adjoint program smaller and faster by factors that can go up to 40%.

- **TBR:** The reverse mode of AD, with the Store-All strategy, stores all intermediate variables just before they are overwritten. However this is often unnecessary, because derivatives of some expressions (e.g. linear expressions) only use the derivatives of their arguments and not the original arguments themselves. In other words, the local Jacobian matrix of an instruction may not need all the intermediate variables needed by the original instruction. The *To Be Restored (TBR)* analysis finds which intermediate variables need not be stored during the forward sweep, and therefore makes the differentiated program smaller in memory.

Several other strategies are implemented in TAPENADE to improve the differentiated code. For example, a data-dependence analysis allows TAPENADE to move instructions around safely, gathering instructions to reduce cache misses. Also, long expressions are split in a specific way, to minimize duplicate sub-expressions in the derivative expressions.

The end-user can specify properties of external or black-box routines. This is essential for real industrial applications that use many libraries. The source of these libraries is often hidden. However AD needs some information about these black-box routines in order to produce efficient code. TAPENADE lets the user specify this information in a separate signature file. The end-user can also improve the generated code via a (too small) number of user directives placed in the source. For instance for the reverse mode, TAPENADE lets the user specify finely which procedure calls must be checkpointed or not,

Figure 3 shows the architecture of TAPENADE. It is implemented mostly in JAVA (115 000 lines) except for the separate front-ends which can be written in any more convenient language. Front- and back-ends communicate with the kernel via an intermediate

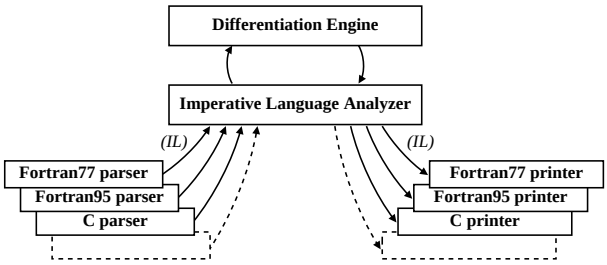

Figure 3: Overall Architecture of TAPENADE

abstract language ("IL") that makes the union of the constructs of individual imperative languages. Notice also the separation between the general-purpose program analysis and the differentiation engine itself.

TAPENADE can be used simply as a web server, available at the URL:
`http://tapenade.inria.fr:8080/tapenade/index.jsp`
It can also be downloaded and installed from our FTP server
`ftp://ftp-sop.inria.fr/tropics/tapenade/README.html`
A documentation is available on our web page
`http://www-sop.inria.fr/tropics/`
and as an INRIA technical report (RT-0300)
`http://hal.inria.fr/inria-00069880`
TAPENADE is distributed with a license, which is free for academic research, and at a price for industrial or commercial use. Several companies have purchased an industrial license for TAPENADE (Rolls-Royce, BAe, Cargill, Credit Suisse), others are in the eval-

uation process. At the same time, TAPENADE is used by many academic institutions for education and research. Many users cannot be identified, because the log files of our web and ftp servers give little information. However, we are aware of TAPENADE regular use by researchers in Argonne National Lab. (Illinois, USA), the Federal Reserve (Washington DC, USA), CSIRO Hobart (Australia), NAL Bangalore (India), Cranfield university (UK), Oxford university (UK), RWTH Aachen (Germany), Humboldt university Berlin (Germany), German Aerospace Center (Germany), DLR (Germany), General Electric Deutschland (Germany), University of Bergen (Norway), ISMAR-CNR Venezzia (Italy), Alenia (Italy), Dassault Aviation (France), INSA Toulouse (France), Université de Montpellier (France), CMAP Ecole Polytechnique (France) ...

Here are some statistics on the use of TAPENADE: There are roughly 2 releases per year. The latest release has been downloaded 123 times from our ftp server, between July 2008 and January 2009 from 23 different countries including Germany (29), France (23), USA (11), Italy (6). The TAPENADE web server has been used more than 5000 times since its creation in 2002 (We count only real uses, no robots). The current rate is 800 uses (sessions) per year. These uses come from about 260 different geographical locations, from 42 different countries including France (48), Germany (33), USA (21), UK (17). More than 100 users gave us their name and application when using the Tapenade web server, and 74 have registered in the "tapenade-users" mailing list.

Competitors to TAPENADE are OPEN-AD, developed mainly at Argonne Nat. Lab., USA, and TAF, developed by FastOpt company in Hamburg, Germany. The three tools are roughly at a similar technical level. There exist many other AD tools that either target a smaller application field or have more restricted functionalities. To our knowledge, there has been no published benchmark. However several users told us they ordered an independent comparative study of the various tools before purchasing a Tapenade license.

During the evaluation period, the team has implemented the following main improvements to TAPENADE:

- We completed the extension to Fortran95.
- We implemented the extension to C, now in the distributed version.
- We implemented and tested pointer analysis.
- The reverse mode now accepts most uses of pointers and allocation.
- We implemented user directives `NOCHECKPOINT` and `II-LOOP`.

3.2 Teaching

L. Hascoët gave lectures on AD in courses organized by ERCOFTAC in 2005 and 2007, and organized a one-day lecture and tutorial during EDF-CEA summer schools in Paris in 2005 and 2006.

L. Hascoët and M. Martinelli organized a one-day tutorial after the NODESIM-CFD conference in Sophia-Antipolis in 2007.

A. Dervieux gave a lecture "Metric-based mesh adaptation" at "Most efficiency in FEM", CIRM session in Marseilles in 2007.

A. Belme, PhD student in TROPICS, gives lectures to 3rd year students of the engineering school of Université de Nice, on Numerical Algorithms. 2 hours a week for 12 weeks in 2009.

3.3 Visibility

A. Dervieux received a prize from the French Academy of Science in 2005.

M. Araya-Polo received a "best student paper" award at the WSEAS conference in Cancun, Mexico in 2005.

L.Hascoët was on the jury for the PhD defense of Claire Lauvernet, on data assimilation from satellite observations, 2005.

A.Dervieux and J.A. Desideri organized the Indian-French Symposium on Modelisation in Aeronautics and Space, in Sophia-Antipolis, 2006.

A.Dervieux and J.Sokolowsky organized a minisymposium on Optimum Design at the CANUM conference in Nice, 2006.

L.Hascoët co-organizes the twice-a-year European AD workshop. Editions were held in Nice in 2005 and Sophia-Antipolis in 2007.

A.Dervieux and J.A. Desideri organized the 42th AAAF Symposium on Multidisciplinary Coupling and Optimization, in Sophia-Antipolis, 2007.

L.Hascoët was on the jury for the HDR defense of Renaud Marlet, on specialization of programs through Partial Evaluation, 2007.

A.Dervieux and F. Beux were editors of a special issue [13] "Shape Design in Aerodynamics, Parameterization and sensitivity" of the European Journal of Computational Mechanics, 2008.

A.Dervieux was on the jury for the PhD defenses of Rémi Bourguet, Raphaël Kuate, and Adrien Loseille, in 2008.

A.Dervieux continues long-term scientific collaboration with Prof Charbel Farhat at Stanford University.

Bruno Koobus, external collaborator of TROPICS, defended his HDR in 2008.

4 External Funding

(k euros)	2005	2006	2007	2008
European projects				
e.g. IST HISAC	24	24	24	24
e.g. IST NODESIM-CFD		54	54	54
Total	24	78	78	78

National initiatives

NEMO: TROPICS participates (for a very small funding) in the CNRS API project "Les Enveloppes Fluides et l'Environnement" (LEFE), started in 2007. LEFE gathers several research groups in Earth Sciences, one part of them working on 4D-Var variational data assimilation. We collaborate with IFREMER Brest, INRA Avignon, IMAG Grenoble and the LOCEAN team in Paris VI University.

EVAFLO: TROPICS participates (for a very small funding) in the ANR project "Evaluation et Validation Automatique pour le calcul FLOttant" (EVAFLO), started in 2007. EVAFLO looks for innovative methods for reliable and certified computation on floating points numbers. AD can be useful to compute intervals and interval approximations of derivatives. Partners are ENS Lyon (main contractor), CEA Saclay, and University of Perpignan.

European projects

HISAC : The main objective of the HISAC -"HighSpeed AirCraft"- IP-project is to establish the technical feasibility of an environmentally compliant supersonic small size transport aircraft (S4TA), through a MultiDisciplinary Optimization (MDO) approach and focused technological improvements. INRIA teams Tropics et Gamma are contributing and have been coordinating (with the help of Dassault) the following tasks: (WP2.3.1) Sonic Boom assessment, and (WP2.3.2) Sonic Boom minimization methods. There are 37 partners, including: DASSAULT AVIATION (FR), ALENIA AERONAUTICA (IT), EADS DEUTSCHLAND (GE), ROLLS ROYCE (UK)...

NODESIM-CFD : "Non-deterministic simulation FOR CFD-based design" is a STREP devoted to handle uncertainties in CFD simulation process by applying non-deterministic methodologies. Output is described by a random variable, instead of a single value. Tools for the evaluation and quantification of uncertainties in aerodynamic and thermal performance predictions are developed, for enhancing design confidence, reducing risk, improving safety. NODESIM-CFD involves INRIA teams TROPICS and OPALE. There are 17 partners, including: NUMECA (BE), AIRBUS-UK (UK), ALENIA (IT), BAE Systems (UK), DASSAULT AVIATION (FR), DLR (GE), ONERA (FR)...

5 Objectives for the next four years

We propose four objectives for the next period, all about Automatic Differentiation. The first two concern the reverse AD model, the third concerns Numerical methodology, and the fourth concerns further development of TAPENADE. Inevitably, these objectives overlap.

- **Data-Flow analysis to improve reverse AD:** This general category of objectives was already present during the last period. We have already proposed a model to combine inversion, recomputation and storage to retrieve control values [16]. We will extend it to general real values. The open problem here is to make the extreme Recompute-All and Store-All strategies get closer or meet. We must also formalize the empirical observation that any checkpointed piece of code must be *reentrant*, i.e. its initial state can be fully stored and restored. This is not always easy to achieve in the case of dynamic allocation or file I-O. Finally, it is high time we do a serious benchmarking comparison of all the methods proposed to improve reverse AD. We can do it because nearly all these methods are now implemented in TAPENADE and we have a large enough collection of real applications. We must do it because the merits of each method have often be measured only vaguely in the past.

- **Reverse AD of MPI or OpenMP parallel programs:** This objective was present at the start of the project, and was delayed until reverse AD of sequential programs is mature enough. This objective is now fully awake [35, 37] because application of reverse AD e.g. to OPA and to the MIT GCM requires it. The open

problem here is to differentiate communication primitives at an elementary level, rather than seeing them as ad-hoc black-boxes. We want to develop a new proof of correctness of our reverse AD of MPI communications primitives, based on the Data-Dependence graph. We also want to implement this model and demonstrate it on real Global Circulation simulators, in collaboration with Earth Sciences researchers. In a similar approach, we want to propose a reverse AD model for OpenMP parallel loops, demonstrate its correctness and validate it on applications. We will work on this objective in collaboration with the teams in Argonne and Aachen, and TROPICS is currently looking for a PhD student to take care of the Numerical aspects.

- **Application to large unsteady applications involving turbulence:** This objective is the natural continuation of the previous periods. Now that we have proposed strategies for steady-state simulations, we will explore the harder case of unsteady simulations. We also want to handle the further degree of complexity that comes when taking into account turbulence. At the same time, we will build on previous work on control of approximation errors (mesh adaptation), and we plan to use gradients now for correction of approximation errors. We will investigate the two types of correctors, direct linearization and Defect Correction on one hand, adjoint-based functional correction on the other hand. Here also, we intend to apply these methods to large unsteady flow simulations. This work is clearly connected to the previous objective, as such large simulations are necessarily parallel. This ambitious objective is the PhD research subject of the new team member Anca Belme. One risk is that, despite our efforts, regular reverse AD of so large codes remains inefficient. If this happens, we will have to explore the variety of approximations and simplifications that hand writers of these adjoints indulge themselves to, and see which of them can be formalized and implemented in the AD tool.

- **Further development of** TAPENADE**:** We must make TAPENADE for C just as robust as for Fortran. This implies a better interface for black-box external procedures and an improved management of standard include libraries. We must also experiment with programs that mix Fortran and C. For the reverse mode, we will add the capacity to checkpoint an arbitrary piece of code designated by the end-user. A more exotic objective is to make TAPENADE a front-end for Adol-C. Adol-C requires the end-user to designate active variables by hand, by modifying their type. TAPENADE can detect active variables and therefore will generate a new source program with the types modified. If time permits at the end of the next period, we might develop a first attempt at differentiating Object-Oriented languages.

6 Bibliography of the project-team

Doctoral dissertations and "Habilitation" theses

[1] M. Araya-Polo, *Approaches to assess Validity of Derivatives and to Improve Efficiency in Automatic Differentiation of Programs*, PhD Thesis, Université de Nice Sophia-Antipolis, 2006.

[2] M. Martinelli, *Sensitivity evaluation in aerodynamic optimal design*, PhD Thesis, Scuola Normale Superiore di Pisa, 2007.

[3] Y. Mesri, *Gestion et contrôle des maillages non structurés anisotropes, applications en aérodynamique*, PhD Thesis, Université de Nice, **co-advised by A. Dervieux**, 2007.

[4] A. Loseille, *Adaptation de maillages anisotropes 3D multi-echelles*, PhD Thesis, Université Pierre et Marie Curie, **co-advised by A. Dervieux**, 2008.

Articles in referred journals and book chapters

[5] M. Araya-Polo, L. Hascoet, "Certification of Directional Derivatives Computed by Automatic Differentiation", *WSEAS Transactions on Circuits and Systems 4*, 4, 2005.

[6] P.-H. Cournède, B. Koobus, A. Dervieux, "Positivity statements for a mixed-element-volume scheme on fixed and moving grids", *REMN Revue Européenne de Mécanique Numérique / European Journal of Computational Mechanics 15*, 7-8, 2006, p. 767–799.

[7] F. Courty, A. Dervieux, "Multilevel functional Preconditioning for shape optimisation", *International Journal of CFD 20*, 7, 2006, p. 481–490.

[8] F. Courty, D. Leservoisier, P.-L. George, A. Dervieux, "Continuous metrics and mesh optimization", *Applied Numerical Mathematics 56*, 2006, p. 117–145.

[9] A. Dervieux, F. Courty, T. Roy, M. Vázquez, B. Koobus, "Optimization loops for shape and error control", *in : Verification and validation methods for challenging multiphysics problems*, B. Bugeda and et al. (editors), CIMNE, Barcelona, 2006, Extended version in INRIA Research Report 5413.

[10] M. Vázquez, A. Dervieux, B. Koobus, "A methodology for the shape optimization of flexible wings", *Engineering Computations 23*, 4, 2006, p. 344–367.

[11] A. Dervieux, Y. Mesri, F. Courty, L. Hascoët, B. Koobus, M. Vázquez, "Calculs de sensibilité par differentiation pour l'Aérodynamique", *ESAIM: PROCEEDINGS 10*, 2007.

[12] F. Alauzet, S. Borel-Sandou, L. Daumas, A. Dervieux, Q. Dinh, S. Kleinveld, A. Loseille, Y. Mesri, G. Rogé, "Multimodel design strategies applied to sonic boom reduction", *REMN Revue Européenne de Mécanique Numérique / European Journal of Computational Mechanics 17*, 1-2, 2008, p. 245–269.

[13] F. Beux, A. Dervieux (editors), *Shape Design in Aerodynamics, Parameterization and sensitivity*, *REMN Revue Européenne de Mécanique Numérique / European Journal of Computational Mechanics*, *17*, 1-2, Hermes, 2008.

[14] A. Dervieux, Y. Mesri, F. Alauzet, A. Loseille, L. Hascoët, B. Koobus, "Continuous mesh adaptation models for CFD", *CFD Journal 16*, 4, 2008, p. 346–355.

[15] L. Hascoët, B. Dauvergne, "Adjoints of large simulation codes through Automatic Differentiation", *REMN Revue Européenne de Mécanique Numérique / European Journal of Computational Mechanics 17*, 63-86, 2008.

[16] L. Hascoët, J. Utke, U. Naumann, "Cheaper Adjoints by Reversing Address Computations", *Scientific Programming 16*, 1, 2008, p. 81–92.

[17] L. Hascoët, M. Vázquez, B. Koobus, A. Dervieux, "A Framework for Adjoint-based Shape Design and Error Control", *CFD Journal 16*, 4, 2008, p. 454–464.

[18] C. Lauvernet, F. Baret, L. Hascoët, S. Buis, F.-X. LeDimet, "Multitemporal-patch ensemble inversion of coupled surface-atmosphere radiative transfer models for land surface characterization", *Remote Sensing of Environment 112*, 3, 2008, p. 851–861.

[19] M. Martinelli, F. Beux, "Multi-level gradient-based methods and parametrization in aerodynamic shape design", *REMN Revue Européenne de Mécanique Numérique / European Journal of Computational Mechanics 17*, 1-2, 2008, p. 169–197.

[20] L. Hascoët, "Reversal Strategies for Adjoint Algorithms", *in : From Semantics to Computer Science. Essays in memory of Gilles Kahn*, Cambridge University Press, 2009, p. 487–503.

Publications in Conferences and Workshops

[21] F. Courty, T. Roy, B. Koobus, M. Vázquez, A. Dervieux, "Error analysis for P1-exact schemes", *in : Finite Element for Flow Problems, Swansea, UK, April 4-6*, 2005.

[22] A. Dervieux, L. Hascoët, M. Vázquez, B. Koobus, "Optimization loops for shape and error control", *in : Recent Trends in Aerospace Design and Optimization*, Tata-McGraw Hill, New Delhi, p. 363–373, 2005. proceedings Post-SAROD-2005, Bangalore, India.

[23] 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, vol. 50*, Springer, 2005.

[24] B. Koobus, L. Hascoët, F. Alauzet, A. Loseille, Y. Mesri, A. Dervieux, "Continuous mesh adaptation models for CFD", *in : Recent Trends in Aerospace Design and Optimization*, Tata-McGraw Hill, New Delhi, p. 3–11, 2005. proceedings SAROD-2005, Hyderabad, India.

[25] C. Lauvernet, F. Baret, L. Hascoët, F.-X. LeDimet, "Improved estimates of vegetation biophysical variables from MERIS TOA images by using spatial and temporal constraints", *in : proceedings of the 9th International symposium on Physical measurements and signatures in remote sensing, ISPMSRS 2005*, 2005.

[26] V. Pascual, L. Hascoët, "Extension of TAPENADE towards Fortran 95", *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, vol. 50*, Springer, 2005.

[27] 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.

[28] M. Fagan, L. Hascoët, J. Utke, "Data Representation Alternatives in Semantically Augmented Numerical Models", *in : 6th IEEE International Workshop on Source Code Analysis and Manipulation, SCAM 2006, Philadelphia, PA, USA*, 2006.

[29] B. Ferron, L. Hascoët, "Capacités actuelles de la Différentiation Automatique: l'adjoint d'OPA par TAPENADE", *in : Colloque National sur l'Assimilation de Données, Toulouse, France*, 2006.

[30] L. Hascoët, M. Araya-Polo, "Enabling User-driven checkpointing strategies in Reverse-mode Automatic Differentiation", *in : Proceedings of the ECCOMAS CFD 2006 conference, Egmond aan Zee, The Netherlands*, 2006. also INRIA Research Report #5930, `https://hal.inria.fr/inria-00079223`.

[31] M. Martinelli, A. Dervieux, L. Hascoët, "Strategies for computing second-order derivatives in CFD design problems", *in : Proc. of West-East High Speed Flow Field Conference*, Moscou, Russia, nov. 19-22 2007.

[32] Y. Mesri, F. Alauzet, A. Dervieux, "Coupling optimum shape design and mesh adaptation for sonic boom reduction", *in : Proc. of 42nd CAA-AAAF*, Sophia-Antipolis (F), march 2007.

[33] Y. Mesri, F. Alauzet, A. Dervieux, "A strongly coupled mesh adaptative optimal shape algorithm", *in : Proc. of NMFD-ICFD*, Reading, UK, march 2007.

[34] M. Martinelli, L. Hascoët, "Tangent-on-Tangent vs. Tangent-on-Reverse for second differentiation of constrained functionals", *in : Advances in Automatic Differentiation*, *Lecture Notes in Computational Science and Engineering, vol. 64*, Springer, 2008. Selected papers from AD2008 Bonn, August 2008.

[35] U. Naumann, L. Hascoët, C. Hill, P. Hovland, J. Riehme, J. Utke, "A Framework for Proving Correctness of Adjoint Message-Passing Programs", *in : Proceedings of the 15th European PVM/MPI Users' Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Springer-Verlag, p. 316–321, 2008.

[36] V. Pascual, L. Hascoët, "TAPENADE for C", *in : Advances in Automatic Differentiation, Lecture Notes in Computational Science and Engineering, vol. 64*, Springer, 2008. Selected papers from AD2008 Bonn, August 2008.

[37] J. Utke, L. Hascoët, C. Hill, P. Hovland, U. Naumann, "Toward Adjoinable MPI", 2009. accepted for PDSEC-09.

Internal Reports

[38] F. Alauzet, Y. Mesri, A. Dervieux, "Sonic Boom reduction by mesh adapted optimal control", *Research Report number European project HISAC 18th Month 2.3*, 2006.

[39] M.-H. Tber, L. Hascoët, A. Vidard, B. Dauvergne, "Building the Tangent and Adjoint codes of the Ocean General Circulation Model OPA with the Automatic Differentiation tool TAPENADE", *Research Report number 6372*, INRIA, 2007, `http://hal.inria.fr/inria-00192415`.

[40] F. Alauzet, Y. Mesri, A. Dervieux, A. Loseille, "Sonic boom simulation and mitigation by mesh adaptation and optimal control", *Deliverable number European project HISAC, 4 reports from 2005 to 2008*, 2008.

[41] M. Martinelli, A. Dervieux, R. Duvigneau, L. Hascoët, V. Pascual, "Management of uncertainties by perturbation and Automatic Differentiation", *Deliverable number European project NODESIM-CFD, 4 reports from 2005 to 2008*, 2008.