Welcome and Introduction to SU2

SU2 WINTER WORKSHOP
FEBRUARY 3RD, 2017

Dr. Thomas D. Economon
Department of Aeronautics & Astronautics
Stanford University

SU2
The Open-Source CFD Code
What is SU2?

The SU2 suite is an open-source collection of C++/Python-based software for multi-physics simulation and design on unstructured meshes (i.e., CFD!).

SU2 is under active development at Stanford University in the Department of Aeronautics and Astronautics and now in many places around the world.

https://github.com/su2code/SU2
http://su2.stanford.edu
Our Guiding Principles

1. Open-source (LGPL 2.1)!
2. Portability and easy installation.
3. Readability, reusability, and encapsulation (C++).
4. Flexibility and automation (Python).
5. High performance.
6. Gradient availability for design, mesh adaptation, UQ, etc.

We believe that an open-source code supported by a large group of developers working in concert has tremendous potential…

• Technical excellence: experts all around the world contribute to produce new research and capabilities not previously envisioned.
• Open, web-based platform encourages global collaboration without geographic limitations.
• Increases the pace of innovation in computational science.
The SU2 Timeline

SUmb solver developed @ ADL

June 2008
Francisco Palacios completes PhD with Juan Alonso on committee

Summer/Fall 2009
Francisco spends 3 months at Stanford

2010
Work on CADES (predecessor to SU2) begins

2011
Preparations for releasing SU2 as open source

2003-2008
2009
2010
2011
“We must think big... on Jan 20th everybody in the aeronautical community must know that there is a new player in the CFD open-source community.”

- Dr. Francisco Palacios, January 9 2012
STANFORD UNIVERSITY UNSTRUCTURED CODE (SU^2) RELEASED TODAY, THURSDAY JANUARY 19, 2012

The First Release of The SU^2 Open-Source Computational Fluid Dynamics (CFD) Analysis and Optimization Suite is Out Today

Stanford University's Aerospace Design Laboratory (ADL) is releasing its Stanford University Unstructured (SU^2) open-source code today, Thursday January 19, 2012. This suite is a collection of C++ based software tools for performing Partial Differential Equation (PDE) analysis and solving PDE constrained optimization problems. The toolset is designed with computational fluid dynamics and aerodynamic shape optimization in mind, but is extensible to treat arbitrary sets of governing equations such as potential flow, electrodynamics, chemically reacting flows, and many others.

A key feature of SU^2 is that it incorporates everything needed to perform a complete design loop; from the ability to compute flow and adjoint solutions, to obtaining objective function sensitivities relative to specified design variables, and, using this information, drive gradient-based shape optimization through built-in mesh deformation algorithms.

The software structure has been designed for maximum flexibility, leveraging the class-inheritance features native to the C++ programming language. This makes SU^2 an ideal vehicle for performing multi-physics simulations. Additionally, the decomposition of the flow solver allows for the rapid implementation of new spatial discretization methods and time-integration schemes.

SU^2 is under active development in the Aerospace Design Lab (ADL) in the Department of Aeronautics and Astronautics at Stanford University and is be released under an open-source license.

If you would like more information on SU^2, please check out http://su2.stanford.edu or email us at susquared-dev@lists.stanford.edu

Jan 17
Pre-release Workshop

Jan 19
SU2 v1.0
SU2 is born!

Jun 25
SU2 v1.1

Oct 31
su2.stanford.edu update

Sept 25
First tweet @su2code
Jan 7
AIAA SciTech Presentation

Jan 8
SU2 v2.0, CFD Online Forum Open

Jan 15
SU2 v2.0 Workshop

May 17 & 25
SU2's first two PhDs

Sept 30
OpenMDAO / SU2 Joint Workshop

Aug 10
SU2 on GitHub

Winter
Spring
Summer
Fall

2013
The Stanford Solar Car Project’s Race for Aerodynamic Efficiency

SU2
The Open-Source CFD Code
Fluid, meet Structure.

You are invited to a live webinar on Thu, Apr 7, 2016, 8:00 AM - 9:00 AM PDT.

The open-source SU2 package for CFD analysis and design was conceived as a tool for multi-disciplinary research. We’ve been hard at work improving our C++ code and more easily support the addition of new physical models and their coupling at a high-level. Today, we are releasing a powerful new example in the form of a fluid-structure interaction (FSI) capability embedded within SU2 v4.2 "Cardinal.

Jan 7
SU2 v4.1
Download SU2 v4.1

Feb 29
NASA LBFD Announced

SU2
The Open-Source CFD Code
Continuous and Discrete.
The open-source SU2 package for CFD analysis and design serves not only as a useful example to computational scientists, but also as a common baseline for future development by the entire community. The current open-source model has enabled the leading experts across many technical areas, anywhere in the world, to work together in creating new capabilities that would not have materialized in the absence of collaboration. Today, we demonstrate this once again with the release of SU2 version 4.1 "Cardinal.

Apr 5
Pointwise, Tecplot, SU Solar Car, SU2 Webinar

SU2 v4.2
Download SU2 v4.2

Jun 15
SU2 v4.2

Jul / Aug
U of Liege Visits Stanford

Aug
SU2 v4.3

Sept 5
1st Annual SU2 Developers Meeting

2016

Winter
Spring
Summer
Fall
SU2 v5.0 (released on January 19, 2017) is just the beginning. Here's a sneak peak at just some of the things we’re working on for the future…
Efficient Aerodynamic Design using the Discrete Adjoint Method in SU2

Tim Albring\(^1\), Max Sagebaum\(^1\) and Nicolas R. Gauger\(^2\)

\(^1\) TU Kaiserslautern, Kaiserslautern, 67663, Germany

\(^2\) Chair for Scientific Computing, TU Kaiserslautern, Bldg 34, Paul-Ehrlich-Strasse, 67663 Kaiserslautern, Germany

I. Introduction

Almost 30 years have passed since Jameson discussed the success and challenges in CFD and listed optimization and design as one of the directions for future research. Unfortunately, computational methods for aerodynamic analysis are even today far from being incorporated in advanced design procedures. Although adjoint methods\(^1\) have greatly decreased the computational effort and impressive results using the continuous and discrete \(^2\) adjoint approaches where published during the last few years, there are still open issues regarding the robustness, (discrete) consistency and generality for complex problems in turbulent flows. Typically, one has to make a compromise between efficiency and the latter properties while choosing an appropriate approach for a given problem. In general there exist different approaches to construct and solve the discrete adjoint system of equations. Most of them require the exact linearization of the flow residual, which is in contrast to the flow solver itself, where some approximation is most of the time sufficient to yield convergence. Depending on the complexity of the numerical methods the linearization by hand is time-consuming and error-prone. Furthermore it lacks the capability of adapting to changes in the flow solver. One way to circumvent this problems is the use of Algorithmic Differentiation (AD) applied to parts of the flow solver\(^3\) to construct the Jacobian. Although it reduces the error-proneness, it still requires the application of AD to all subroutines involved in the computation of the residual. Even if we have the exact Jacobian, it is typically ill-conditioned so that applying a Krylov-method can be inefficient. This is often visible when including turbulence models. Here, Duality-Preserving methods can be useful as they guarantee to have the same convergence as the originally computed by Korivi and Newman\(^4\), albeit called Incremental Iterative Form. Until today they are only used by a manageable amount of people.\(^5\)

In this paper we want to show that by exploitation of the fixed-point structure of the flow solver it is possible to derive a duality-preserving iteration to solve the adjoint system. All occurring gradients can be constructed by applying AD to the top-level routine of the flow solver, thereby eliminating the computational cost of the Jacobian matrix. Furthermore, we apply advanced AD techniques like expression templates\(^6\) and local preaccumulation\(^7\) to automatically generate a representation of the computational graph of each expression at compile-time. This results in competitive performance while still maintaining flexibility. Due to the use of AD the extension to new turbulence models, transition models, fluid models or objective functions is straightforward. To contribute to the open-source idea, we therefore tightly integrated the discrete adjoint solver along with the AD features into the open-source framework SU2.\(^8\) In order for the community to explore new and interesting optimization problems. For this reason we also dedicate a section to show a short overview on the implementation.

---

1. Introduction

In the past decades significant progresses in the numerical methods for the design and optimization of aircraft have been made. Since the advent of adjoint-based methods,\(^9\) for which the computational cost is independent of the design variables, researchers have been able to tackle many large-scale and practical problems. For aero and aerostuctural optimizations of complete aircraft configurations.\(^10\) In most applications, the adjoint method is considered to be a steady state, as evident from the rich body of literature on the use of the adjoint method for optimization of turbulence models, transition models, fluid models or objective functions. The intent is to make it straightforward to extend the AD system to a large number of people.\(^5\)

In order to accurately account for the complex mesh movement in the governing equations, adjoint equations introduces additional difficulties. With the growth of computing power and the improvement of time-accurate numerical methods, has led to a reduction in the use of adjoint-based methods. The renewed interest in reducing aircraft noise and emissions, due to the ever-increasing aviation noise regulations, also served as a necessary push and pull for the development of the adjoint method for use in aerodynamic design optimization.
Lines of Code in SU2 by Release (w/out comments or blanks)

- **C/C++**
- **C/C++ Header**
- **Python**
- **Total**

*includes code in externals/*
Where are we today? Everywhere.

Since release in Jan 2012: ~500k web visits across 182 countries.

10s of thousands of downloads, 11,000+ addresses on the users email list.

240+ forks on GitHub, 170+ addresses on the developers email list.

Webhits at su2.stanford.edu. Data as of 2017.02.03.
Traffic data from the SU2 GitHub repository (accessed 2017.02.03).
Multiple options…

1. Check out the download portal on the main website to register and download binaries or source for v5.0.0:

   http://su2.stanford.edu/download.html

2. Download releases (latest and older) from GitHub here:

   https://github.com/su2code/SU2/releases

3. **Recommended**: Clone the open repository directly at the command line to get the latest release. Note: the master branch (default) is always stable:

   $ git clone https://github.com/su2code/SU2.git

**SU2 v5.0.0 “Raven”** was released on January 19, 2017, the 5th anniversary of SU2.
A large body of documentation is available!

Main documentation found on the SU2 GitHub wiki linked on the main page under “Guides”:
- [https://github.com/su2code/SU2/wiki](https://github.com/su2code/SU2/wiki)
- Detailed information on installation, input and output files, etc.
- Contains step-by-step tutorials.
- Wiki-style docs that the developers maintain.
- See current contents to the right…
Documentation / Tutorials

• Additional training materials: http://su2.stanford.edu/training.html
  • Links to tutorials, presentations, files, and videos.

• Active forum on CFD Online: http://www.cfd-online.com/Forums/su2/

• AIAA publications with many technical details on physics, numerical methods, and V&V:
SU2 Winter Workshop
Feb 3rd, 2017
13:00 – 16:00, PST
Stanford, CA 94305

Meeting Agenda

Part I
13.00 – 13.15: Welcome & Introduction
13.15 – 13.35: Tutorial 1: Basic Analysis & Configuration Options
   Running SU2 & familiarization with analysis options & capabilities.
13.35 – 13.45: Q&A
13.45 – 14.05: Tutorial 2: Python Scripts & Optimization Problems
   Advanced features of SU2, inputs to the SU2 python scripts.
14.05 – 14.15: Q&A
14.15 – 14.30: Coffee Break

Part II
   Understanding how SU2 works & how to modify it.
14.45 – 14.50: Q&A
14.50 – 15.00: Introduction to Github and SU2 Development Best Practices
   How to share your changes to SU2 with the world.
15.00 – 15.45: Interactive Exercise: Modifying a Python Script
   Recommended: bring an idea of a problem that require running several small CFD solutions sequentially – for example a sweep of an input parameter or uncertainty quantification.
15.45 – 16.00: Open Discussion

In order to participate (in-person or virtually), please register for the meeting by following the link included in the email announcement.
Thank you for your interest in SU2. Please make sure to install SU2 and run at least one tutorial prior to the workshop. (See https://github.com/su2code/SU2/wiki)

To find more information about SU2 or to get involved, please visit the following pages:
- SU2 on GitHub: https://github.com/su2code/SU2
- SU2 Forum on CFD Online: http://www.cfd-online.com/Forums/su2/
- Follow SU2 on Twitter: https://twitter.com/su2code
Institutions that have downloaded SU2. Sized by frequency.