A practical review on linear and nonlinear global approaches to stability analysis

D. Fabre[(a)] **et al.**
IMFT, University of Toulouse

*This paper is aimed to review the linear and nonlinear approaches to study the stability of fluid flows. We provide a concise but self-contained exposition of the main concepts and specific numerical methods designed for global stability studies, including the classical linear stability analysis, the adjoint-based sensitivity and the most recent nonlinear developments. A simple computer code, that implements all the concepts discussed in the present paper, is provided. In particular, we discuss the mesh adaptation, the use of simple shift-invert strategy instead of the classical Arnoldi algorithm and a reformulation and simplification of the recent nonlinear self-consistent approach proposed by Mantič-Lugo et al [**?**] in terms of a harmonic balance. The code is demonstrated for the reference case of the incompressible, two-dimensional flow around a circular cylinder, but the software is easily customisable to a variety of other flow configurations or flow equations.*

1 Introduction

The concept of stability bears on the reaction of a system to a small perturbation of its state. If the generic disturbance grows in time, the system is unstable. The concept of stability can be simply formulated for a system of Ordinary Differential Equations (ODE). Such systems can be at equilibrium, where the state does not depend on time, or can present a periodic state, with all components returning to the same values, after every period.

The stability of fluid flows usually depends on the value of a given parameter. A bifurcation occurs when a critical value is reached and the original solution becomes linearly unstable, with the system tending towards a new steady or unsteady state. In the second part of the 19th century, specific analytical and numerical methods able to study these bifurcations have emerged and are continuously evolving up to the present days. A crucial point, that drove the development in this field, is the availability of larger and larger computing resources. Initially, the linear stability theory focused on fluid flows that are homogeneous in two spatial directions, e.g. plane Poiseuille flow [**?**]. This implies that the streamwise and the spanwise base-flow gradients vanish and, as a consequence, it is possible to consider only one (streamwise) velocity component. Thus, the stability problem requires a *local* numerical resolution, i.e. a one-dimensional problem. On the other hand, when there are at least two spatial variables, the class of methods suited to solve such problems are generally called "global stability approaches". A classical example of such behavior in fluid dynamics is the instability occurring in the wake of a circular cylinder. At low Reynolds number (precisely for $Re < 46.7$) the flow is steady and symmetric, but for larger values of Re a global instability arises in the flow field leading to the well-known von Kármán vortex alley. This flow configuration has served as a benchmark in the development of this class of methods. If one is only interested in predicting the stability or instability of a flow, it is enough to conduct a *linear stability analysis* which is the fundamental brick of global stability approaches. Beyond this simple question, in the past two decades, a number of extensions have been developed and popularized. *Adjoint methods* are an important extension [**?**], [**?**] ; they can give insight into the sensitivity of the flow to intrinsic or extrinsic contributions. *Nonlinear stability approaches* [?], [**?**] have also been developed in order to extend the range of applicability of the approach towards large amplitude perturbations.

The objective of the present work is to contribute to the popularization of such methods in two ways:

> First, we give a concise but self-contained exposition of the main concepts and specific numerical methods pertaining to global stability, including basic linear stability, adjoint-based sensitivity, as well as the most recent nonlinear developments.
> Secondly we offer an open-source and user-friendly software called "StabFem" [1] to perform such calculations. The software combines program written in both FreeFem++ and Matlab languages. FreeFem++ is used to generate and adapt the meshes and to solve the various linear problems arising in the analysis. Matlab is used as a driver to monitor the computations, perform the required loops over parameters, and plot the results.

In the present paper the concepts are introduced and the software is demonstrated for the reference case of the incompressible, two-dimensional flow around a cylinder, but the

[1] The StabFem software may be obtained at the following url : https://github.com/erbafdavid/StabFem

software is easily customizable to a variety of other situations (compressible, three-dimensional, etc..).

Although we don't claim to invent any radically new method, our exposition and implementation contains a number of originalities making the computation particularly efficient in terms of computational time and memory (all the figures of the paper can be produced in only a few minutes on a standard laptop). The most notable originalities are the systematic use of mesh adaptation (§2 and 3), the use of simple shift-invert instead of Arnoldi (§3), and a reformulation and simplification of the nonlinear self-consistent approach of Mantic-Lugo et al in terms of a harmonic balance (§4).

2 Linear stability analysis : equations and methods

2.1 Computing a base-flow with Newton iteration

Navier-Stokes equations and weak form We start from the general problem of a flow field $[\mathbf{u}, p]$ satisfying the incompressible Navier-Stokes equations on a domain Ω,

$$\partial_t \mathbf{u} = \mathcal{NS}(\mathbf{u}; p) \equiv -\mathbf{u}\cdot\nabla\mathbf{u} - \nabla p + \frac{2}{Re}\nabla\cdot\mathsf{D}(\mathbf{u}) \qquad (1)$$

$$\nabla\cdot\mathbf{u} = 0, \qquad (2)$$

with suitable boundary conditions on the frontier $\partial\Omega$ of the domain. Here $\mathsf{D}(\mathbf{u})$ is the rate-of-strain tensor defined as

$$\mathsf{D}(\mathbf{u}) = 1/2\left(\nabla\mathbf{u} + \nabla^T\mathbf{u}\right)$$

In the framework of finite element methods, we need to write the equation in weak form. Prior to this we define a scalar product as follows, for either scalar or vectorial quantities $\langle\phi_1, \phi_2\rangle$:

$$\langle\phi_1, \phi_2\rangle = \int_\Omega \bar{\phi}_1 \cdot \phi_2 \, \mathrm{d}\Omega$$

The weak form of the Navier-Stokes equations is readily defined by introducing test functions $[\mathbf{v}, q]$ associated with the momentum and continuity equations, and integrating over the domain[2]

$$\forall[\mathbf{v}, q], \quad \partial_t \langle\mathbf{v}, \mathbf{u}\rangle = \left\langle\mathbf{v}, \mathcal{NS}(\mathbf{u}; p)\right\rangle + \langle q, \nabla\cdot\mathbf{u}\rangle . \qquad (3)$$

Newton iteration We look for a steady base-flow $(\mathbf{u}_b; p_b)$ satisfying the steady Navier-Stokes equations, i.e. $\mathcal{NS}(\mathbf{u}_b, p_b) = 0$. Suppose that we have a 'guess' for the base flow $[\mathbf{u}_b^g, p_b^g]$ which almost satisfies the equations. We look for a better approximation under the form

$$[\mathbf{u}_b, p_b] = [\mathbf{u}_b^g, p_b^g] + [\delta\mathbf{u}_b, \delta p_b]. \qquad (4)$$

Injecting (4) into the weak form (3) of the Navier-Stokes equations and developing up to linear terms in terms of the perturbation lead to $\mathcal{NS}(\mathbf{u}_b^g, p_b^g) + \mathcal{L}_{\mathbf{u}_b^g}(\delta\mathbf{u}_b, \delta p_b; \mathbf{u}_b^g)$, which can also be written in weak form as :

$$\begin{aligned} &\left\langle\mathbf{v}, \mathcal{NS}(\mathbf{u}_b^g)\right\rangle + \left\langle q, \nabla\cdot\mathbf{u}_b^g\right\rangle \\ &+ \left\langle\mathbf{v}, \mathcal{L}_{\mathbf{u}_b^g}(\delta\mathbf{u}_b, \delta p_b; \mathbf{u}_b^g)\right\rangle + \langle q, \nabla\cdot\delta\mathbf{u}_b\rangle = 0, \end{aligned} \qquad (5)$$

where $\mathcal{L}$ is the linearised Navier-Stokes operator, defined by its action on a flow field $(\mathbf{u}; p)$ as follows

$$\mathcal{L}_{\mathbf{U}}(\mathbf{u}, p; \mathbf{U}) = -\mathcal{C}(\mathbf{U}, \mathbf{u}) - \nabla p + \frac{2}{Re}\nabla\cdot\mathsf{D}(\mathbf{u}), \qquad (6)$$

and $\mathcal{C}$ is the convection operator defined by

$$\mathcal{C}(\mathbf{U}, \mathbf{u}) = (\mathbf{U}\cdot\nabla)\,\mathbf{u} + (\mathbf{u}\cdot\nabla)\,\mathbf{U}. \qquad (7)$$

This problem can now be discretized by projecting upon a basis of Taylor-Hood $(u, v, p) \to (P2, P2, P1)$ finite elements. Noting δX the discretization of $(\delta\mathbf{u}_b; \delta p_b)$ this eventually leads to a matricial problem of the form $A\cdot\delta X = Y$. The procedure of Newton iteration is to solve iteratively this set of equations up to convergence. In our implementation, the algorithm is written in the Freefem++ solver *Newton_2D.edp* which is wrapped by the Matlab driver *Freefem_BaseFlow.m*.

2.2 Linear stability

Direct eigenvalue problem We study the onset of the instability within the linear theory by using a normal-mode analysis:

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_b(\mathbf{x}) + \varepsilon\hat{\mathbf{u}}(\mathbf{x})e^{\lambda t}, \;\; p(\mathbf{x}, t) = p_b(\mathbf{x}) + \varepsilon\hat{p}(\mathbf{x})e^{\lambda t} \qquad (8)$$

where $\lambda = \sigma + i\omega$ is the eigenvalue, σ the amplification rate, ω the oscillation rate, $\hat{\mathbf{u}}, \hat{p}$ the eigenmodes, and ε a small parameter. The eigenmodes and the eigenvalues are the solution of the following eigenproblem:

$$\lambda\hat{\mathbf{u}} = \mathcal{L}_{\mathbf{u}_b}(\hat{\mathbf{u}}, \hat{p}), \qquad (9)$$

or, in weak form :

$$\lambda\langle\mathbf{v}, \hat{\mathbf{u}}\rangle = \left\langle\mathbf{v}, \mathcal{L}_{\mathbf{u}_b}(\hat{\mathbf{u}}, \hat{p})\right\rangle + \langle q, \nabla\cdot\hat{\mathbf{u}}\rangle . \qquad (10)$$

[2] In the simple presentation given here we have omitted the issue of boundary conditions. Details on way boundary conditions can be incorporated in the weak formulation through integration by parts can be found in the appendix.

After discretization, we end up with an eigenvalue problem with the matricial form

$$\lambda B\hat{X} = A\hat{X} \tag{11}$$

Where A is the matrix resulting from the discretization of $\mathcal{L}_{\mathbf{u}_b}$, i.e. the same matrix appearing in the Newton computation of the base flow, and B is a 'weight' matrix associated to the scalar product $\langle v,u\rangle = \int \overline{\mathbf{v}}\cdot\mathbf{u}\,\mathrm{d}\Omega$.

Adjoint eigenvalue problem and structural sensitivity Developed in the two past decades, the concept adjoint modes has now become an unavoidable complement to the linear global stability approach. We here give a short summary of the definition and usefulness of this concept, we refer to Luchini & Bottaro [**?**] for further details. First of all, the *adjoint linearised Navier-Stokes operator* $\mathcal{L}^\dagger$ is defined thanks to the following property:

$$\begin{aligned}\forall(\mathbf{u},p;\mathbf{v},q),\ \left\langle \mathcal{L}^\dagger_{\mathbf{U}}(\mathbf{v},q),\mathbf{u}\right\rangle + \langle\nabla\cdot\mathbf{v},p\rangle \\ = \langle\mathbf{v},\mathcal{L}_{\mathbf{U}}(\mathbf{u},p)\rangle + \langle q,\nabla\cdot\mathbf{u}\rangle\,.\end{aligned} \tag{12}$$

We can then define the adjoint eigenvalues and eigenmodes as the solutions of the eigenvalue problem

$$\forall(\mathbf{u},p),\quad \lambda^\dagger\langle\hat{\mathbf{v}},\mathbf{u}\rangle = \left\langle \mathcal{L}^\dagger_{\mathbf{U}}(\hat{\mathbf{v}},\hat{q}),\mathbf{u}\right\rangle + \langle\nabla\cdot\hat{\mathbf{v}},p\rangle\,. \tag{13}$$

It can be shown [**?**] that the adjoint eigenvalues $\lambda_k^\dagger$ are the complex conjugates of the direct eigenvalues λ_k.

Although the concept of adjoint operator may sound complicate, the resolution of the adjoint problem using finite elements methods is actually extremely easy. In effect, the scalar product used in the definition of the weak formulation and that appearing in the definition of adjoint being the same, the weak formulations of both problems are thus identical when exchanging the test functions and the unknown functions. Thus the matricial form of the discretized version of (13) is deduced from the one of the direct problem by a simple (Hermitian) transpose of the matrix :

$$\overline{\lambda}^\dagger B\hat{X}^\dagger = A^T\hat{X}^\dagger. \tag{14}$$

Adjoint eigenmodes are a powerful tool for investigating problems such as receptivity, transient growth, control and sensitivity (see the reviews of [**?**], [**?**], [?]). The simplest physical interpretation of an adjoint eigenmode is as follows : it corresponds to the initial condition which has maximum projection along the direction of the corresponding eigenmode. Thus, the adjoint of the most amplified mode corresponds to the optimal perturbation which will maximize the growth of energy in the limit of large time. In effect, one can prove that for $t\to\infty$ the asymptotic behaviour of a solution with initial condition $\mathbf{u}_i$ is given as :

$$\mathbf{u}(t) \approx \frac{\langle\hat{\mathbf{u}}^\dagger,\mathbf{u}_i\rangle}{\langle\hat{\mathbf{u}}^\dagger,\hat{\mathbf{u}}\rangle} e^{\lambda t}\hat{\mathbf{u}}.$$

The choice $\mathbf{u}_i = \hat{\mathbf{u}}^\dagger$ is the initial condition of norm unity which maximizes the first factor in this expression.

The adjoint eigenmode also allows us to introduce the so-called *structural sensitivity tensor* that is defined as

$$\mathbf{S}(\mathbf{x}) = \frac{||\hat{\mathbf{u}}^\dagger||\,||\hat{\mathbf{u}}||}{\langle\hat{\mathbf{u}}^\dagger,\hat{\mathbf{u}}\rangle}, \tag{15}$$

which has became popular in the recent years. This quantity is a direct and practical measure of the effect of perturbations of the linear operator on the eigenvalue. The region of the flow where $\mathbf{S}(\mathbf{x})$ reaches its maximum values is, thus, the region where the instability mechanism originates, and is often referred to as the *wavemaker region*.

Iterative methods for eigenvalue computations When it comes to the numerical resolution of generalized eigenvalue problems such as $AX = \lambda BX$ (or its adjoint version (14)), several methods are possible. Direct methods to compute the whole spectrum are both costly prohibitive and useless. A popular alternative is the use of iterative methods which allow us to compute a limited set of eigenvalues located in the vicinity of a "shift" value λ_{shift}. The simplest version of this method is the simple shift-invert iteration, which consists of solving iteratively the system

$$X^n = (A-\lambda_{shift}B)^{-1}BX^{n-1}.$$

It is easy to show that this iterative procedure quickly asymptotes to $X^{n+1}\approx(\lambda^{*-1})^n\hat{X}$ where $\hat{X}$ is the eigenmode with largest λ^{*-1} (i.e. the one with eigenvalue λ closest to the shift).

When a good estimation of the eigenvalue is available, this method converges very rapidly and is very efficient, but it can only provide a single eigenvalue. If we want to compute a larger number of eigenvalues, we can revert to a generalized version of iterative methods, called Arnoldi methods [**?**]. The shift-invert version of the Arnoldi method is in fact the most commonly used method of the current time and is at the basis of both the popular matlab function `eigs` and the eigenvalue solver of FreeFem (i.e. ARPACK++). Our implementation in StabFem allows to chose between single eigenvalue computation (power method) and multiple eigenvalue computation (Arnoldi). The selection is made according to the parameter "nev" transmitted to the driver.

2.3 Mesh adaptation procedure

As for any numerical method, a crucial point in the numerical efficiency is the design of the mesh. The finite element method allows to use unstructured mesh and hence to locally adapt the refinement. The most common procedure is to decompose the domain into several parts with different grid densities; for instance for the wake of a cylinder, we will design a near-wall region with very small size, a "wake" region with intermediate mesh size, and an outer region with

```
baseflow = SF_Init('Mesh_Cylinder_Large.edp');
baseflow = SF_BaseFlow(baseflow,'Re',1);
baseflow = SF_BaseFlow(baseflow,'Re',10);
baseflow = SF_BaseFlow(baseflow,'Re',60);
baseflow = SF_Adapt(baseflow,'Hmax',10,'InterpError',0.01);
[ev,em] = SF_Stability(baseflow,'shift',0.04+0.74i,'type','S')
baseflow = SF_Adapt(baseflow,em,'Hmax',10,'InterpError',0.01);
plotFF(baseflow,'mesh'); plotFF(baseflow,'ux');

Re_Range = [2 : 2: 50]; Drag_tab = []; Lx_tab = [];
    for Re = Re_Range
        baseflow = SF_BaseFlow(baseflow,'Re',Re);
        Drag_tab = [Drag_tab,baseflow.Drag];
        Lx_tab = [Lx_tab,baseflow.Lx];
    end
plot(Re_Range,2*Drag_tab,'b+-');
plot(Re_Range,Lx_tab,'b+-');

Re_Range = [40 : 2: 100];lambda_branch=[];
baseflow=SF_BaseFlow(baseflow,'Re',40);
[ev,em] = SF_Stability(baseflow,'shift',-.03+.72i,'nev',1,'type','D');
    for Re = Re_Range
        baseflow = SF_BaseFlow(baseflow,'Re',Re);
        [ev,em] = SF_Stability(baseflow,'nev',1,'shift','cont');
        lambda_branch = [lambda_branch ev];
    end
plot(Re_Range,real(lambda_branch),'b+-');
plot(Re_Range,imag(lambda_branch)/(2*pi),'b+-');
```

Fig. 1. Illustration of the usage of the StabFem software to produce an adapted mesh and study the base flow and the linear stability properties of a cylinder (from script *SCRIPT_CYLINDER_DEMO.m*)

large mesh size. The inconvenient is that the design relies on an a priori expectation of the regions where gradients will be large.

In our implementation, we used an automatic mesh adaptation method. The implementation relies on the AdaptMesh procedure of the FreeFem++ software. This procedure is detailed in detail in ref. [**?**]. In short, the classical Delaunay-Voronoi algorithm produces a mesh with grid-point distribution specified by a *Metric* matrix $\mathcal{M}$. The AdaptMesh algorithm consists of using as a metric the *Hessian* (second-order spatial derivatives) of an objective function u_h defined over the domain, i.e. $\mathcal{M} = \nabla\nabla u_h$. The precision can be controlled by specifying an objective value for the interpolation error of the function on the new mesh.

To build an optimal mesh for the base-flow calculation, the idea is to use as the objective function u_h the solution $\mathbf{u}_b$ itself, as computed on a previous mesh.The base flow is then recomputed on the adapted mesh, providing a better approximation of the solution. The procedure can be repeated a few steps to ensure a right convergence.

The mesh generated in the previous way may not be optimal for the stability calculations as the structure of the eigenmode may be more complex than that of the base flow. To remedy with this, the idea is to subsequently adapt the mesh to both the mesh flow and the results of the stability calculation. This is easily done with FreeFem, as the AdaptMesh procedure can be used with several objective functions. We have experimented two different ideas. The first is to adapt the mesh to the base flow and the structure of the leading eigenmode. A second and smarter idea is to adapt the mesh to the base flow and the structural sensitivity. Results show that the two procedures yield the same values for the eigenvalues with less than 1% error, but that the second method gives a mesh with about 3 times less grid points, leading to much faster computations (see appendix A). So, if one is interested only in the eigenvalues, for instance to plot growth rates as function of Reynolds or to identify the critical Reynolds number, mesh adaptation to the sensitivity gives the best option. On the other hand, if one is interested in the detailed structure of the eigenmode, which may extend far away from the wavemaker region, it is advised to use mesh adaptation on the eigenmode.

In our implementation, the whole process is performed using the Matlab driver *Freefem_AdaptMesh.m* ; the kind of adaptation (to base flow only, to eigenmode, or to sensitivity) is decided by the choice of parameters transmitted to this function.

3 Illustration for the wake of a cylinder

Problem description Here, we consider the two-dimensional flow past a circular cylinder. All flow quantities are normalized using the uniform incoming velocity U_∞ and the cylinder diameter D, which are the characteristic

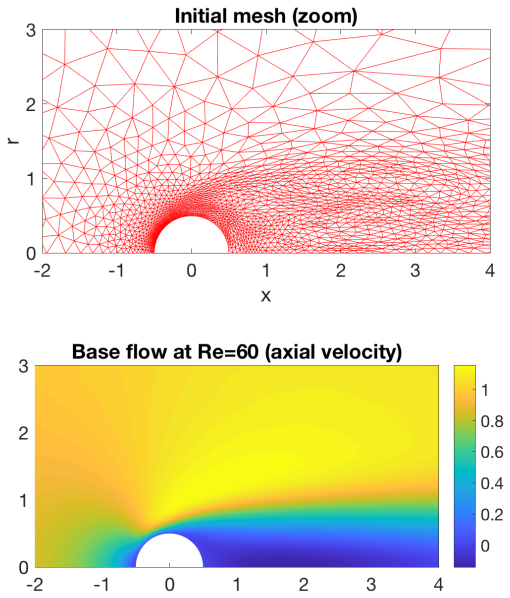

Fig. 2. Adapted mesh (top) and base flow (axial velocity component) $(bottom)$ for the flow over a cylinder at $Re = Re_c = 46.7$.

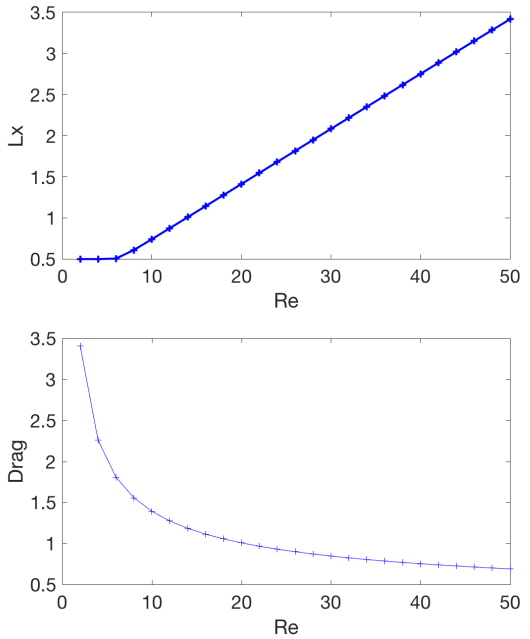

Fig. 3. Recirculation length L_x (a) and drag C_x (b) of the base flow over a cylinder as function of Re.

velocity and length scales used for the definition of Reynolds number $Re = U_\infty D/\nu$. The origin of the cartesian frame of reference is considered located on the cylinder axis, the x-axis is chosen to be parallel to the incoming free-stream velocity while the y-axis with the cross-stream velocity. The dimensions of the computational domain are the following: $-40 \le x/D \le 80$ and $-40 \le y/D \le 40$. We impose a no-slip condition $(u = 0, v = 0)$ on the cylinder surface (Γ_{cil}), a uniform velocity $(u = 1, v = 0)$ at the inflow $x/D = -40$ and a no-stress condition at the outlet $x/D = 80$ and on the lateral boundaries of the computational domain $y/D = \pm 40$.

The hydrodynamic loads can be obtained by integrating the stress tensor over the cylinder surface. In particular, the hydrodynamic lift and drag forces read

$$D = F_x(\mathbf{u}, p) = \int_{\Gamma_{cil}} \left[-p\mathbf{n} + \frac{2}{Re}\mathbf{D}(\mathbf{u}) \cdot \mathbf{n} \right] \cdot \mathbf{e_x} d\ell \qquad (16)$$

$$L = F_y(\mathbf{u}, p) = \int_{\Gamma_{cil}} \left[-p\mathbf{n} + \frac{2}{Re}\mathbf{D}(\mathbf{u}) \cdot \mathbf{n} \right] \cdot \mathbf{e_y} d\ell. \qquad (17)$$

Mesh adaptation procedure Let's consider now the matlab code reported in figure 1. First we build an initial mesh (line 1), and compute base flow solutions for increasing values of the Reynolds number up to $Re = 60$ (lines 3-9). Then we perform the mesh adaptation with the structural sensitivity, as explained in a previously (line 13). The resulting mesh, depicted in figure 2, is used for the rest of the computations presented in this paper (except for displaying the eigenmode in figure 4a). Appendix A presents additional test regarding mesh convergence, and demonstrate that results obtained with the resulting mesh are trustable within 1% accuracy for the eigenvalue, while requiring a very reasonable number of grid points compared to previous studies of this problem.

Base flow Having thus produced a convenient mesh, we can now illustrate the properties of the base flow as function of Reynolds number. Figure 1 shows how to compute and plot with StabFem the two most commonly studied quantities, namely the recirculation length $Lx(Re)$, i.e. the location of the stagnation point at the rear of the recirculation region, and the drag coefficient $C_x(Re)$. Note that the object `baseflow` is defined as a structure with fields `Drag` and `Lx`. The resulting plots are given in figure 3, and are in good agreement with known results for this classical problem. In particular, for low Reynolds, the recirculation $L_x(Re)$ is equal to 0.5 (which is the radius of the cylinder) indicating the absence of a recirculation region. The latter appears for $Re > 4.8$, in accordance with known results.

Linear stability results We investigate the stability of the base flow fields by performing a parametric study of the eigenproblem (11). In this way, we determine the critical Reynolds number Re_c at with the steady base flow first becomes unstable: to this end it is useful to remember that a flow state is linearly unstable when the real part of the leading eigenvalue, i.e. the growth rate, is positive. Figure 4 shows growth rate and the Strouhal number $St = a\omega/2\pi U_\infty$ as a function of the Reynolds number. It is easy to check that the critical Reynolds number is about 47 for the first mode. The associated direct eigenmode is depicted in figure

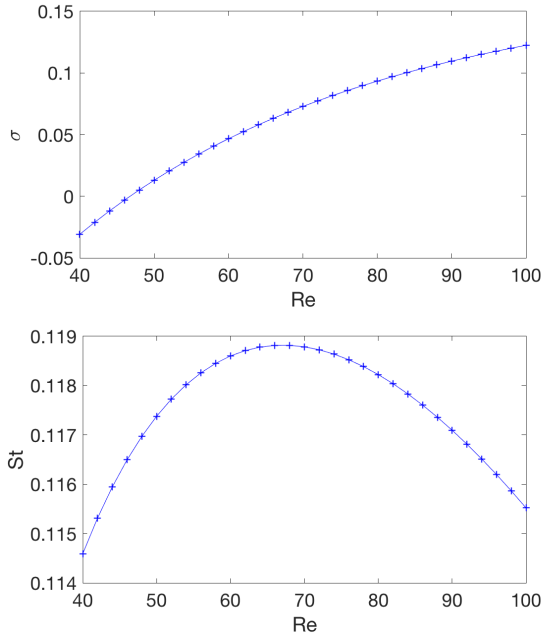

Fig. 4. Growth rate σ (a) and Strouhal number $St = (a\omega)/(2\pi U_\infty)$ (b) as function of Reynolds number

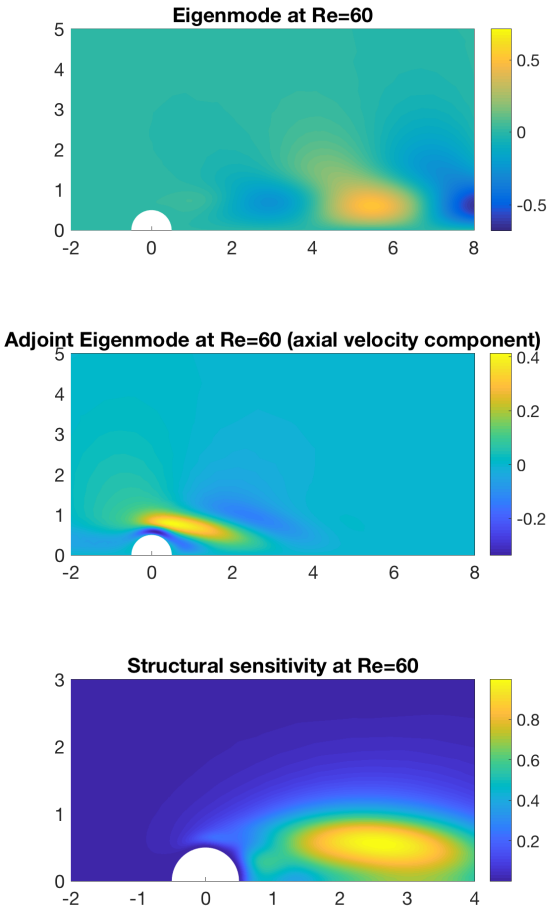

Fig. 5. Contour plot of the streamwise velocity component: (a) (Direct) Eigenmode; (b) Adjoint mode and (c) Structural sensitivity field for the cylinder's wake at $Re = Re_c = 46.7$.

5. The spatial structure of this mode extends downstream of the bluff body and is characterized by streamwise extended spatial disturbances. On the other hand, the adjoint mode is highly localized near the cylinder on the upper (and lower) side of the body surface. We recall that the adjoint field provides useful information about the mechanism to flow receptivity to momentum forcing and mass injection. We note also that this receptivity decays rapidly both upstream and downstream of the bluff body.

The structural sensitivity field displayed in figure 5c) is obtained by evaluating eq. 15. In particular, we plot the spectral norm of the sensitivity tensor, it gives the maximum possible coupling among the velocity components. This spatial map can be used to identify the flow region where the instability mechanism acts.

4 Nonlinear global stability approaches

We turn now to analyze how nonlinear effects can be considered in the framework of global stability analysis. Here, we symbolically write the Navier-Stokes system as $\partial_t \mathbf{u} = \mathcal{NS}(\mathbf{u}; p)$, therefore dropping the systematic reference of the incompressibility constraint and associated pressure field.

Review on nonlinear global stability approaches In the past decade, efforts have been devoted to extend the range of validity of global approaches into the nonlinear regime for $Re > Re_c$, with the double objective to describe the properties of the limit cycle reached after saturation and to derive amplitude equations describing the transitent dynamics towards this cycle. The first milestone in this direction is the work of Sipp & Lebedev [?] who used a weakly nonlinear development in terms of the distance to the threshold $Re - Re_c$. However, although derived on rigorous grounds, the weakly nonlinear development has a very limited range of validity and already fails for $Re \geq 47$. More recently, Mantič-Lugo et al [?] proposed an alternative approach termed *Self-consistent model* which succeeds in predicting the dynamics up to $Re = 120$ and more. This model was build as a linear stability approach build on a *mean flow* which includes a retroaction of the eigenmode. However, if one is interested in the limit cycle and not the transient, the approach is actually equivalent to a Fourier series decomposition of the flow with respect to time (an approach also known as *harmonic balance*) retaining only the two first terms.

In the next sections, we thus successively present the weakly nonlinear model, the self-consistent model and eventually the harmonic-balance model, and propose a direct resolution method.

Weakly nonlinear model We give here a brief account of the weakly nonlinear model of Sipp & Lebedev [?], also discussed by Gallaire et al. [**?**]. The derivation is based on a multiple scale decomposition based on the parameter

```
1  [baseflow,em]=SF_FindThreshold(baseflow,em);
2  wnl = SF_WNL(baseflow);
3
4  Re_HB = [Rec 47.5 48 49 50 55 60 65 70 75 80 85 90 95 100];
5  omega_HB = [Omegac]; Aenergy_HB = [0];
6
7  baseflow=SF_BaseFlow(baseflow,47.5);
8  [ev,em] = SF_Stability(baseflow,'shift',Omegac*i);
9  Astart = sqrt(wnl.Lambda/(wnl.Nu0+wnl.Nu2)*(1/Rec-1/47.5));
10 [meanflow,mode] = SF_HarmonicBalance(baseflow,em,'sigma',0.,'Re',47.5,'Aguess',Astart);
11
12 for Re = Re_HB(2:end)
13     [meanflow,mode] = SF_HarmonicBalance(meanflow,mode,'Re',Re);
14     omega_HB = [omega_HB imag(mode.lambda)];
15     Aenergy_HB  = [Aenergy_HB mode.Energy];
16 end
```

Fig. 6. Illustration of the procedure for nonlinear calculations using StabFem (from script *SCRIPT_CYLINDER.m*.

$\varepsilon^2 = 1/Re - 1/Re_c$, and the slow time scale $\tau = \varepsilon t$. They started by considering the following asymptotic expansion of the flow state $\mathbf{q} = (\mathbf{u}, p)$:

$$\begin{aligned}\mathbf{q} = \mathbf{q}_{bc} + \varepsilon\left[A_{wnl}(\tau)\hat{\mathbf{q}}e^{i\omega_c t} + c.c.\right] \\ + \varepsilon^2\left[\mathbf{q}_\varepsilon + |A_{wnl}|^2\mathbf{q}_{2,0} + \left(A_{wnl}^2\mathbf{q}_{2,2}e^{2i\omega_c t} + c.c.\right)\right] + O(\varepsilon^3)\end{aligned} \tag{18}$$

where $\mathbf{q}_{bc}$ is the base flow at the threshold Re_c, $\hat{\mathbf{q}}$ the neutral eigenmode at Re_c, $A_{wnl}(\tau)$ a slowly varying amplitude. Substituting the above expansion (18) into the Navier-Stokes equations and grouping terms multiplied by the same power of ε, a hierarchy of equations is obtained. The order ε^2 contains three terms respectively computed as the solutions of the following linear problems :

$$\mathcal{L}_{\mathbf{u}_{bc}}(\mathbf{u}_\varepsilon) + 2\nabla\cdot\mathbf{D}(\hat{\mathbf{u}}) = 0, \tag{19}$$

$$\mathcal{L}_{\mathbf{u}_{bc}}(\mathbf{u}_{2,0}) = \mathcal{C}(\hat{\mathbf{u}}, \overline{\hat{\mathbf{u}}}), \tag{20}$$

$$\mathcal{L}_{\mathbf{u}_{bc}}(\mathbf{u}_{2,2}) - 2i\omega_c\mathbf{u}_{2,2} = \frac{1}{2}\mathcal{C}(\hat{\mathbf{u}}, \hat{\mathbf{u}}). \tag{21}$$

Finally, compatibility conditions are imposed at order ε^3 leading to an amplitude equation with the following form :

$$\frac{\partial A_{wnl}}{\partial \tau} = \Lambda A_{wnl} - (\nu_0 + \nu_2)|A_{wnl}|^2 A_{wnl}, \tag{22}$$

where

$$\Lambda = \frac{\left\langle \hat{\mathbf{u}}^\dagger, (\mathcal{C}(\mathbf{u}_\varepsilon, \hat{\mathbf{u}}) - 2\nabla\cdot\mathbf{D}(\hat{\mathbf{u}}))\right\rangle}{\langle \hat{\mathbf{u}}^\dagger, \hat{\mathbf{u}}\rangle}, \tag{23}$$

$$\nu_0 = -\frac{\left\langle \hat{\mathbf{u}}^\dagger, \mathcal{C}(\mathbf{u}_{20}, \hat{\mathbf{u}})\right\rangle}{\langle \hat{\mathbf{u}}^\dagger, \hat{\mathbf{u}}\rangle}, \tag{24}$$

$$\nu_2 = -\frac{\left\langle \hat{\mathbf{u}}^\dagger, \mathcal{C}(\mathbf{u}_{22}, \overline{\hat{\mathbf{u}}})\right\rangle}{\langle \hat{\mathbf{u}}^\dagger, \hat{\mathbf{u}}\rangle}. \tag{25}$$

If one is interested in the amplitude of the limit cycle, we have $|A_{wnl}| = \sqrt{\Lambda_r/(\nu_{0,r} + \nu_{2,r})}$ and eventually, reintroducing the scaling :

$$|A| = \varepsilon|A_{wnl}| = \frac{\Lambda_r}{\nu_{0,r} + \nu_{2,r}}\sqrt{\frac{1}{Re_c} - \frac{1}{Re}}. \tag{26}$$

The frequency of the limit cycle is given by

$$\omega = \omega_c + \left(\Lambda_r - \Lambda_i\frac{\nu_{0,i} + \nu_{2,i}}{\nu_{0,r} + \nu_{2,r}}\right)\left(\frac{1}{Re_c} - \frac{1}{Re}\right). \tag{27}$$

Figure 6, which is extracted from the script `SCRIPT_CYLINDER_DEMO.m`, illustrate the sequence of commands to perform the weakly nonlinear study for the cylinder wake. On line 1, we first determine the instability threshold, and the corresponding base flow and eigenmode[3]. On line 2, we then compute all the terms and coefficients of the WNL model.

As can be seen on figure 7, the approach gives good prediction in the immediate vicinity of Re_c, but deviation appears very rapidly and disagreement is already large at $Re = 48$ (although, as discussed by Gallaire et al.? and also observed by Tchoufag et al. [**?**], a convenient definition of the ε improves the results).

Self-Consistent approach We now briefly review the self-consistent approach as introduced by Mantič-Lugo et al [?]. The authors adopted a pseudo-eigenmode expansion of the flow as follows:

$$\mathbf{u} = \mathbf{u}_m + A_{sc}\left[\tilde{\mathbf{u}}_1 e^{\sigma_{sc}t + i\omega_{sc}t} + \tilde{\mathbf{u}}_{-1}e^{\sigma_{sc}t - i\omega_{sc}t}\right], \tag{28}$$

[3]This routine uses Newton iteration to directly compute the base flow, the eigenmode, the frequency and the critical Reynolds. The algorithm is very similar to the one presented for the HB, with an additional unknown (Re) and an additional constraint (normalization of the mode). The interested reader should reconstruct easily the whole procedure from the code provided.

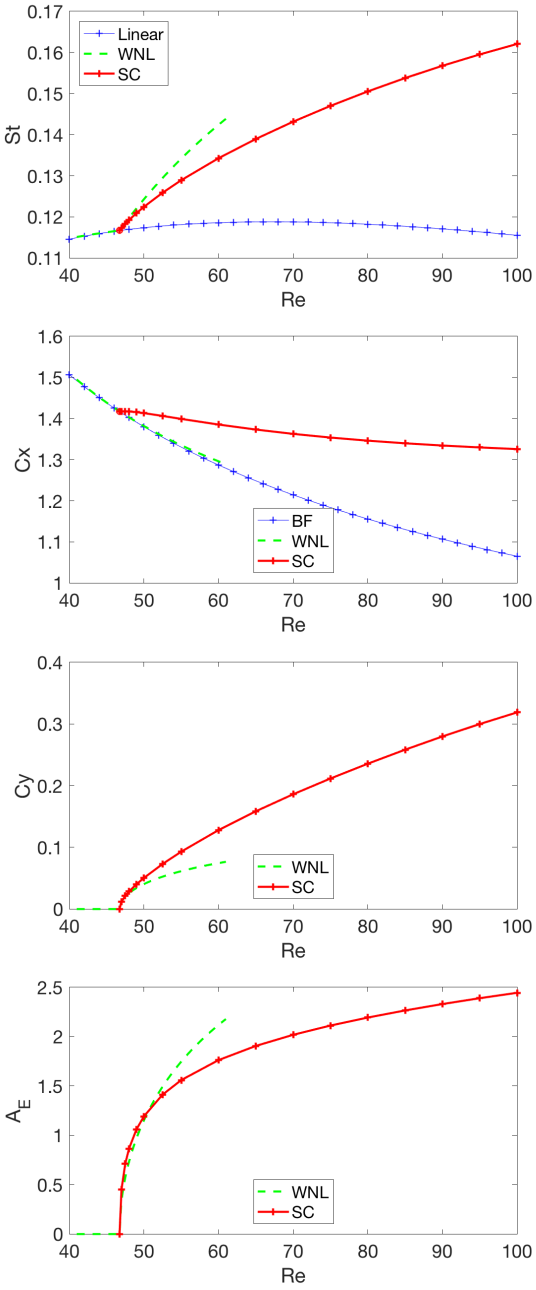

Fig. 7. Comparison between the weakly nonlinear results (WNL) and the harmonic balance data (HB). We report also the Strouhal number computed by using the classical linear stability theory (Linear) and the base flow (BF) C_x.

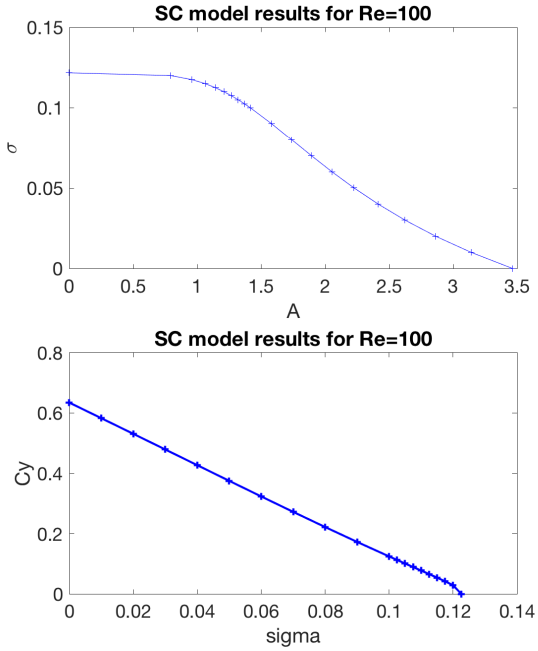

Fig. 8. Self-consistent approach for the wake of a cyrcular cilinder: results at $Re = 100$.

where $\mathbf{u}_m$ is the *mean flow* defined by phase-averaging over the cycle, $\tilde{\mathbf{u}}_1$ is a pseudo-eigenvector which is normalized by the condition $||\tilde{\mathbf{u}}_1|| = 1/\sqrt{2}$, A_{sc} is an amplitude parameter directly related to the energy of the oscillating flow, and $\lambda_{sc} = \sigma_{sc} + i\omega_{sc}$ is a pseudo-eigenvalue which depends upon the parameter A_{sc}. Introducing the decomposition (28) into the Navier-Stokes equations we are left with the following problem:

$$\mathcal{NS}(\mathbf{u}_m) - A_{sc}^2 C(\tilde{\mathbf{u}}, \bar{\tilde{\mathbf{u}}}) = 0, \tag{29a}$$

$$(\sigma_{sc} + i\omega_{sc})\tilde{\mathbf{u}} = \mathcal{L}_{\mathbf{u}_m}(\tilde{\mathbf{u}}). \tag{29b}$$

Equation (29a) provides the mean flow field $\mathbf{u}_m$ while the pseudo-eigenpairs $(\lambda_{sc}, \tilde{\mathbf{u}})$ can be computed by solving the eigenvalue problem (29b).

The self-consistent model has the following properties :

- For $A \ll 1$ it is equivalent to the linear eigenvalue problem (9), and the generalized eigenvalue coincides with the one predicted by linear stability : $\sigma_{SC} + i\omega_{SC} = \sigma_{lin} + i\omega_{lin}$.
- For $\sigma_{SC} = 0$ (corresponding to a specific choice of the amplitude $A = A_c$), the expansion (28) is equivalent to a Fourier decomposition of the limit cycle with only two term retained.
- For $0 < A < A_c$, the resolution leads to a relation $\sigma_{SC}(A); \omega_{SC}(A)$ such that $0 < \sigma_{SC}(A) < \sigma_{lin}$. Although in this case the expansion (28) cannot represent the flow for all t, Mantič-Lugo et al [?] argued that the relation between σ_{SC} and A can be used to build an amplitude equation which captures the transient approach to the limit cycle.

Figure 8 shows the results of the self-consistent model at Re=100.

Harmonic balance : equations and procedure As discussed above, the Self-Consistent model was initially presented by the authors as a rational way to explain the dynamics in terms of a stability analysis of the *mean flow*. Further-

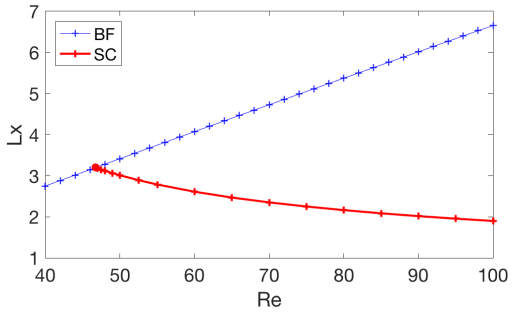

Fig. 9. Flow past a circular cylinder: length of the wake bubble (measured from the rear stagnation point) for different Reynolds number. Comparison between the base flow bubble and the mean flow bubble computed by using the harmonic balance.

more, the resolution method presented by the authors, which involves a double iteration loop, is not easy to implement and not particularly efficient.

If one is only interested by the properties of the limit cycle and not the transient, the model is actually equivalent to a Fourier series, and amenable to a more direct resolution method. We thus start with a Fourier decomposition of the limit cycle under the form :

$$\mathbf{u} = \mathbf{u}_0 + \mathbf{u}_1 e^{i\omega t} + \mathbf{u}_{-1} e^{-i\omega t}. \tag{30}$$

Here $\mathbf{u}_0$ is the mean flow (averaged over time), $\mathbf{u}_1$ is the mode (Fourier component at the fundamental frequency), $\mathbf{u}_{-1} = \overline{\mathbf{u}_1}$ is its complex conjugate, and ω is the (real) oscillation frequency of the limit cycle (which is not known a priori). Injecting this ansartz into the Navier-Stokes equations and taking the mean value and the first Fourier component leads to the following coupled equations :

$$\mathcal{NS}(\mathbf{u}_0) - \mathcal{C}(\mathbf{u}_1, \overline{\mathbf{u}}_1) = 0, \tag{31}$$

$$i\omega\mathbf{u}_1 = \mathcal{L}_{\mathbf{u}_0}(\mathbf{u}_1). \tag{32}$$

Note that contrary to the SC model, the $\mathbf{u}_1$ component is not normalized. The amplitude A can easily be computed a posteriori as $A = \sqrt{2||\mathbf{u}_1||}$. On the other hand, the Fourier description of the limit cycle is invariant with respect to the phase ϕ (i.e. the equations are unchanged when replacing $[\mathbf{u}_1, \mathbf{u}_{-1}]$ by $[\mathbf{u}_1 e^{i\varphi}, \mathbf{u}_{-1} e^{-i\varphi}]$). To allow resolution we thus need an extra equation to fix this phase. Several choices are possible, but a convenient one is to set the lift associated with the imaginary part of $\mathbf{u}_1$ to lift. Physically, this means that the instant $t = 0$ will correspond to a maximum of lift. This condition reads :

$$\Im\{F_y(\mathbf{u}_1)\} = 0. \tag{33}$$

Mathematically, the system (31,32,33) for the unknown $[\mathbf{u}_0, \mathbf{u}_1, \omega]$ is well posed, and can be solved using a Newton iteration just as explained for the base flow in section 2. To allow resolution, we separate the mode into real and imaginary part, by writing $\mathbf{u}_1 = \mathbf{u}_{1,r} + i\mathbf{u}_{1,i}$. We then assume that we know a 'guess'

$$[\mathbf{u}_0, \mathbf{u}_{1,r}, \mathbf{u}_{1,i}, \omega] = [\mathbf{u}_0^g, \mathbf{u}_{1,r}^g, \mathbf{u}_{1,i}^g, \omega^g] + [\delta\mathbf{u}_0, \delta\mathbf{u}_{1,r}, \delta\mathbf{u}_{1,i}, \delta\omega].$$

Injecting and developing up to linear order leads to the following

$$\begin{aligned}
&\mathcal{NS}(\mathbf{u}_0^g) - [\mathcal{C}(\mathbf{u}_{1,r}^g, \mathbf{u}_{1,r}^g) + \mathcal{C}(\mathbf{u}_{1,i}^g, \mathbf{u}_{1,i}^g)] \\
&+ \mathcal{L}_{\mathbf{u}_0^g}(\delta\mathbf{u}_0) - 2[\mathcal{C}(\mathbf{u}_{1,r}^g, \delta\mathbf{u}_{1,r}) + \mathcal{C}(\mathbf{u}_{1,i}^g, \delta\mathbf{u}_{1,i})] = 0; \\
&\mathcal{L}_{\mathbf{u}_0^g}(\mathbf{u}_{1,r}^g) - \omega^g \mathbf{u}_{1,i}^g \\
&+ \mathcal{L}_{\mathbf{u}_0^g}(\delta\mathbf{u}_{1,r}^g) - \mathcal{C}(\delta\mathbf{u}_0^g, \mathbf{u}_{1,r}) - \delta\omega^g \delta\mathbf{u}_{1,i}^g - \omega^g \delta\mathbf{u}_{1,r} = 0; \\
&\mathcal{L}_{\mathbf{u}_0^g}(\mathbf{u}_{1,i}^g) + \omega^g \mathbf{u}_{1,i}^g \\
&+ \mathcal{L}_{\mathbf{u}_0^g}(\delta\mathbf{u}_{1,i}^g) - \mathcal{C}(\delta\mathbf{u}_0^g, \mathbf{u}_{1,i}) + \delta\omega^g \delta\mathbf{u}_{1,i}^g + \omega^g \delta\mathbf{u}_{1,r} = 0; \\
&F_y(\mathbf{u}_{1,i}^g) + F_y(\delta\mathbf{u}_{1,i}) = 0.
\end{aligned} \tag{34}$$

After discretization, this leads to a linear problem of the form $AX = Y$ where X is the discretized version of the unknowns $[\delta\mathbf{u}_0, \delta\mathbf{u}_{1,r}, \delta\mathbf{u}_{1,i}, \delta\omega]$, and A is a matrix of dimension $3N_{dof} + 1$.

As mentioned before, figure 7 shows the comparison between the weakly non linear approach and the harmonic balance data. Figure 9 compares the length of the cylinder bubble computed by using the harmonic balance with the base flow length.

5 Conclusion

The aim of the present review is to introduce advanced stability approaches from a `practical` point of view. The reader can reproduce all the figures presented in this paper by just running the Matlab code available in the StabFem repository.

Acknowledgements

bla bla

Appendix A : details on mesh convergence

In this appendix we compare the results obtained with 8 different meshes, to demonstrate the efficency of the mesh adaptation process. The whole results can be obtained using the script *SCRIPT_CYLINDER_MESHCONVERGENCE.m* available in the StabFem repository.

Giannetti & Luchini [?] showed that the accuracy of the stability results for the flow past a circular cylinder is strictly related to the mesh characteristics in the wavemaker region, i.e. the region where the structural sensitivity reaches its

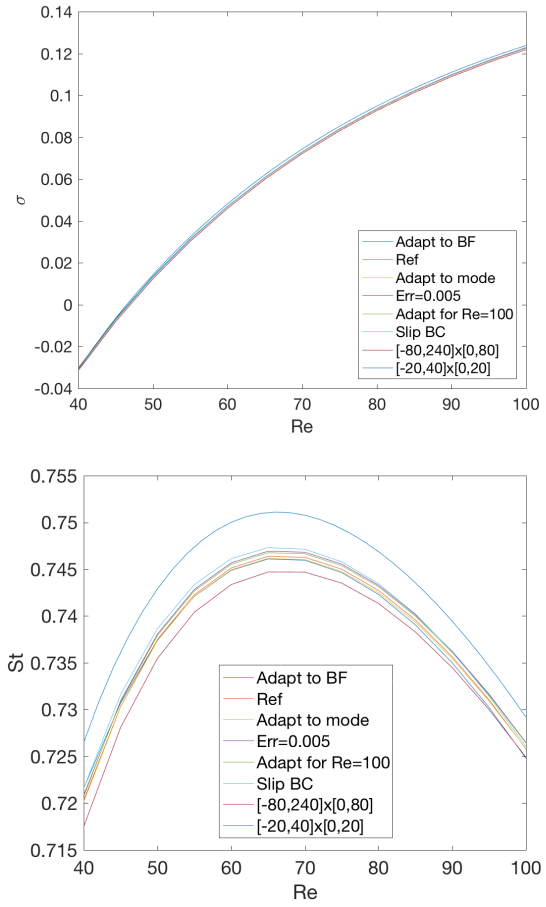

Fig. 10. Growth rate σ (a) and Strouhal number $St = (a\omega)/(2\pi U_0)$ (b) as function of Reynolds number FOR 8 DIFFERENT MESHES

Mesh	N_p	$N_{d.o.f}$	Dimensions	h_{min}	h_{wall}	h_{wake}
BaseFlow	1468	12856				
Ref.	2055	18105				
Mode	12308	109788				
Err= 0.005	4355	38559				
Slip B.C.						
Larger						
Smaller						

Table 1. Description of the 8 different meshes used in the convergence study

maximum values. Following this idea, we chose as reference solution the field computed on an adapted mesh obtained by using the structural sensitivity field. We set the interpolation error equal to 0.01 and the size of the computational domain is $[-40,120]\times[0,40]$. The base flow is computed by using classical no-slip boundary conditions on the body surface, a uniform velocity profile at the inlet and no-stress boundary conditions at the outlet and lateral boundaries. The conditions for the stability computations are simply derived from the ones of the base flow.

Figure 10 displays global stability results. We note that there is a very weak variation in the growth rate and we found an error up to 1% on the Strouhal number. Figure 11 illustrate the procedure adopted to compute the base flow and the adapted mesh. Figure 12 reports the Matlab code adopted to adapt the mesh by using the information provided by both the base flow and the sensitivity field.

6 Detail of the weak formulation

SECTION TO BE COMPLETED.

We have to explain the integration by parts of the viscous term.

$$\forall(\mathbf{v};q), \tag{35}$$
$$\int\left[\mathbf{v}\cdot\mathcal{C}(\mathbf{u}_b^g,\delta\mathbf{u}_b)+\nabla\cdot\delta\mathbf{u}_b q-\nabla\cdot\mathbf{v}\delta p_b+\tfrac{2}{Re}\mathbf{D}(\delta\mathbf{u}_b):\mathbf{D}(\mathbf{v})\right]$$
$$+\int\left[\mathbf{v}\cdot(\mathbf{u}_b^g\cdot\nabla\mathbf{u}_b^g)+\nabla\cdot\mathbf{u}_b^g q-\nabla\cdot\mathbf{v}p_b^g+\tfrac{2}{Re}\mathbf{D}(\mathbf{u}_b^g):\mathbf{D}(\mathbf{v})\right]=0$$

```
> baseflow=SF_Init('Mesh_Cylinder_Large.edp');
      ### INITIAL MESH CREATED WITH np = 2207 points
> baseflow=SF_BaseFlow(baseflow,'Re',1);
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 1
      # Base flow converged in 6 iterations ; Drag = 5.9876; Lx = 0.501
> baseflow=SF_BaseFlow(baseflow,'Re',10);
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 10
      # Base flow f converged in 5 iterations ; Drag = 1.4519; Lx = 0.73191
> baseflow=SF_BaseFlow(baseflow,'Re',60);
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 60
      # Base flow converged in 6 iterations ; Drag = 0.66348; Lx = 3.6989

> baseflow=SF_Adapt(baseflow,'Hmax',10,'InterpError',0.01);
     ### ADAPT mesh to base flow ; InterpError = 0.01  ; Hmax = 10
      #   Number of points np = 1352 ; Ndof = 11862
      #  h_min, h_max : 0.018824 , 10.2895
      #  h_(A,B,C,D) : 0.027467 , 0.28644 , 0.45948 , 0.89245
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 60
      # Base flow converged in 5 iterations ; Drag = 0.65805; Lx = 4.0737

> plotFF(baseflow,'mesh');
```

Fig. 11. Illustration of the procedure to compute a base flow and an adapted mesh using StabFem (from script *SCRIPT_CYLINDER.m*.

```
> [ev,em] = SF_Stability(baseflow,'shift',0.04+0.76i,'nev',1,'type','D');
> [baseflow,em]=SF_Adapt(baseflow,em,'Hmax',10,'InterpError',0.005);
      ### ADAPT mesh to base flow AND MODE  ; InterpError = 0.005  ; Hmax = 10
      #   Number of points np = 13458 ; Ndof = 120030
      #  deltamin, deltapax : 0.0089891 , 13.367
      #  delta_(A,B,C,D) : 0.020536 , 0.082852 , 0.07316 , 0.11914
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 60
      # Base flow converged in 4 iterations ; Drag = 0.64695; Lx = 4.0715
      ### FUNCTION SF_Stability : computation of 1 eigenvalues/modes (DIRECT) with FF solver
      # Stability calculation converged in 4 iterations, lambda = 0.047152+0.74808i.

> [ev,em] = SF_Stability(baseflow,'shift',0.04+0.76i,'nev',1,'type','S');
> [baseflow,em]=SF_Adapt(baseflow,em,'Hmax',10,'InterpError',0.005);
      ### ADAPT mesh to base flow AND SENSITIVITY ; InterpError = 0.005  ; Hmax = 10
      #   Number of points np = 4355 ; Ndof = 38559
      #  deltamin, deltapax : 0.0058536 , 13.1164
      #  delta_(A,B,C,D) : 0.01049 , 0.091861 , 0.13319 , 0.89372
      ### FUNCTION SF_BaseFlow : computing base flow for Re = 60
      # Base flow converged in 3 iterations ; Drag = 0.64758; Lx = 4.0725
      ### FUNCTION SF_Stability : computation of 1 eigenvalues/modes (DIRECT) with FF solver
      # Stability calculation converged in 3 iterations (D+A+S), lambda = 0.047165+0.74823i.
```

Fig. 12. Illustration of the procedure to adapt the mesh for stability calculations with StabFem (from script *SCRIPT_CYLINDER.m*.

Mesh	N_v	N_{ddl}	δ_{min}	δ_{max}	δ_A	δ_B	δ_C	δ_D
$\mathbf{M}_0$(Adapt on base flow)	1429		0.0131	14.33	0.0259	0.514	0.819	1.067
$\mathbf{M}_1$(Adapt on sensitivity)	2038		0.0155	14.17	0.02826	0.2046	0.3909	1.2014
$\mathbf{M}_2$(Adapt on sensitivity, split)	7974		0.0077	7.1	0.014	0.1	0.2	0.6
$\mathbf{M}_3$(Adapt on mode)	12080		0.00825	124.61	0.0229	0.143	0.0993	0.0934
$\mathbf{M}_4$(Adapt on adjoint)	11564		0.0091	14.14	0.0228	0.130	0.109	0.0796

Table 2. Description of meshes used for validation of mesh adaptation strategy

Mesh	L_x	F_x	λ	$L_{x,SC}$	$F_{x,SC}$	ω_{SC}	$F_{Y,SC}$	$A_{E,SC}$
$\mathbf{M}_0$	4.0733	0.6431	$0.047056+0.74416i$	2.6073	0.69277	0.84334	0.1298	1.8301
$\mathbf{M}_1$	4.075	.0.64348	$0.046719+0.74489i$	2.	0.69246	0.84351	0.12789	1.7134
$\mathbf{M}_2$	4.0772	0.64391	$0.04684.+0.74511i$	2.6141	0.69338	0.84383	0.12799	2.048
$\mathbf{M}_3$	4.0748	0.64402	$0.04676+0.74502i$	2.613	0.690338	0.84371	0.12809	2.5905
$\mathbf{M}_4$	4.0772	0.64384	$0.046857+0.74511i$	2.6146	0.6932	0.84333	0.1276	2.5971

Table 3. Results for mesh adaptation strategy ($Re = 60$)

Mesh	N	L_x	F_x	λ	$L_{x,SC}$	$F_{x,SC}$	ω_{SC}	$F_{Y,SC}$	$A_{E,SC}$
$\mathbf{M}_2$(ref)	7974	4.0772	0.64391	$0.04684.+0.74511i$	2.6141	0.69338	0.84383	0.12799	2.048
$\mathbf{M}_5[-20,40]x[0,20]$	7437	4.108	0.64987	$0.048368.+0.74952i$	2	0.69957	0.85015	0.13125	2.3638
$\mathbf{M}_6[-80,160]x[0,80]$	7480	4.0641	0.64129	$0.046271+0.74341i$	2	0.69069	0.84159	0.12653	2.4453
$\mathbf{M}_7$(slip conditions)	7056	4.0467	0.66295	$0.049216+0.76553i$	2	0.69241	0.8438	0.12716	2.3202

Table 4. Results for mesh adaptation strategy ($Re = 60$)