POLITECNICO DI TORINO

CORSO DI LAUREA IN INGEGNERIA AEROSPAZIALE

CHARACTERIZATION OF A TURBULENT CHANNEL FLOW USING COMPLEX NETWORKS

CARATTERIZZAZIONE DI UN FLUSSO TURBOLENTO TRAMITE RETI COMPLESSE

Relatore:
Ing.
STEFANIA SCARSOGLIO

Candidato:
FRANCESCO FIUSCO

Luglio 2016

Anno Accademico 2015/2016

. . .

Sommario

Lo studio della turbolenza è sempre stato un ramo molto attivo della fluidodinamica. Ciò è dovuto al fatto che è osservabile nella vita quotidiana, sia in fenomeni naturali (correnti atmosferiche e oceaniche, diffusione del fumo, flusso cardiocircolatorio) che in applicazioni (flusso in tubazioni, flusso attorno ad un profilo alare, turbomacchine). Nonostante generalmente sia un fenomeno dissipativo (e pertanto indesiderato), in determinate situazioni si rivela utile: basti pensare al mescolamento di fluidi (molto più efficace quando turbolento), alla trasmissione del calore e all'uso dei vortex generator sulle ali di impiego aeronautico per anticipare la transizione turbolenta, che presenta il vantaggio di ritardare lo stallo.

Nonostante sia così facilmente osservabile, la turbolenza continua a essere uno dei grandi problemi irrisolti della fisica moderna, a causa dell'intrinseca non linearità delle equazioni che la descrivono.

Gli approcci moderni generalmente prevedono l'utilizzo di tecniche numeriche (computazionalmente costose) ed esperimenti successivamente analizzati con metodi statistici. In entrambi i casi si ha una gran quantità di dati spesso difficili da interpretare. Man mano che la potenza di calcolo é andata aumentando, questo problema si é reso sempre più pressante, portando alla necessità di un approccio multidisciplinare che potesse semplificare questo tipo di analisi. In questo contesto, la teoria delle reti complesse - un'applicazione della teoria dei grafi - rappresenta un metodo per caratterizzare sistemi complessi che includono un grande numero di entità interagenti tra di loro.

La teoria delle reti complesse é stata utilizzata con buoni risultati in svariati ambiti, dalla sociologia alla biologia, dall'informatica alla progettazione urbana; negli ultimi anni sono stati proposti alcuni approcci basati su di essa per lo studio di turbolenze, approcci in grado di fornire un'idea di base del sistema in oggetto senza ricorrere a tecniche più costose.

In particolare, in questa tesi si userà una tecnica basata sulle reti complesse per studiare

il flusso turbolento in un canale, utilizzando dei dati disponibili online (e ottenuti tramite simulazione DNS) sul JHTDB (John Hopkin's Turbulence Database); in sostanza, si analizzeranno le differenze tra reti ottenute utilizzando i dati prelevati da diverse regioni del flusso.

Contents

Introduction **1**

1 General properties of Turbulence **3**
1.1 Kolmogorov hypotheses[6] 5

2 Complex network theory **8**
2.1 Main definitions and notations 8
2.2 Metric properties 9

3 The visibility algorithm **12**

4 Results of the simulation **15**
4.1 The JHTDB database 15
4.2 Characteristics of the flow 16
4.3 The networks 17
4.4 Discussion of the results 20

Conclusions **22**

Bibliography **23**

Appendix **25**

List of Figures

1.1 Graphic representation of the scales introduced 7

3.1 Visibility algorithm 13

4.1 Energy over time 17
4.2 Network of the buffer layer point 18
4.3 Network of the external region point 18

Introduction

The study of turbulence has always been a very interesting subject in fluid dynamics. This is due to the fact that it is commonly observable in many everyday situations, both in natural phenomena (atmospheric currents, diffusion of smoke, sea currents, blood flow in the circulatory system) and industrial applications (flow of a fluid through pipes, aeronautic applications, turbomachinery). Despite usually being a dissipative, unwanted phenomenon, there is a variety of situations in which turbulent transition is deliberately induced: a common example is the mixing of fluids, that is much more efficient in a turbulent flow, or the transmission of heat, let alone the usage of vortex generators on airplane wings in order to delay stall by anticipating turbulent transition.

Even if its applications are so common, turbulence still represents one of the most challenging problems of nowadays physics, and a general, analytical approach for dealing with turbulent problems is yet to be found (because of the high non-linearity of the equations describing it, as explained later). Feynman himself described turbulence as '*the most important unsolved problem of classical physics'*.

Modern approaches usually involve numeric techniques (very expensive, from a computational point of view) and experimental simulations, that are later treated with statistical methods. Both ways provide a huge amount of data, that are often difficult to be examined and interpreted. With increasing computational power, this problem has become more and more important, leading to the need of interdisciplinary techniques that can make statistical analysis lighter. In this situation, complex network theory - a combination of graph theory and large-number statistic - can be a powerful method to analyse very complex systems with a great number of entities interacting dynamically, just as turbulent flows.

Complex network theory has been successfully used in many branches of science, going from sociology (study of social networks) to biology, computer science, biochemistry, optimization problems in urban design, providing promising results. In the last few years,

only a few complex network approaches have been proposed to study turbulent flows, mainly to give a preliminary spatial characterization of a turbulent flow. In this context, these approaches can represent an extremely efficient tool to obtain a basic overview of the general properties of a turbulence.

In particular, in this work we will apply a network-based approach to a turbulent flow in a channel. We will use time series data obtained via a DNS simulation in a database freely available online (John Hopkins Turbulence Database JHTDB) in different regions of the flow. We will then use an algorithm (called *visibility algorithm*) to obtain a complex network for each point and compare their different topology in order to gather some information about the underlying physical system, and evaluate which the properties of the time series are retained by the complex network. In particular, the thesis will be organized as follows:

- In Chapter 1 we will give a general overview of turbulent flows; we will outline the dynamic processes that influence their behaviour and the difficulties encountered in dealing with turbulent problems;
- Chapter 2 is about the main features of complex networks: some basic notions about graph theory and the main parameters that characterize the topology of a network;
- Chapter 3 is dedicated to the visibility algorithm, as proposed by Lacasa *et al.*[4];
- Chapter 4 deals with the analysis and comparison of the complex networks obtained in various points of the turbulent flow, discussing their differences and how the dynamic system from which they are generated influences their properties; in particular, the points will have the same x and z coordinates, i.e. we will move along the direction normal to the channel wall (because it is the inhomogeneous one, hence the most interesting to study).

Some closing remarks are discussed in the Conclusions section. Finally, all the MATLAB scripts devised to perform the analysis are reported in Appendix.

Chapter 1

General properties of Turbulence

Every fluid system can be described with a set of equations called *Navier-Stokes equations*, which in the general case look like the following (in their differential form):

$$\begin{cases} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \\ \rho(\frac{\partial \vec{u}}{\partial t} + u \nabla u) = \rho F_v + \nabla \cdot \underline{\mathrm{T}} \end{cases}$$

The former equation represents the conservation of mass, the latter is the quantity of motion balance. The nature of the flow depends on the value of some parameters (mainly the Reynolds number): in particular, when the Reynolds number reaches a certain threshold (which depends on the phenomenon we are observing), we have the so called *transition from laminar to turbulent flow*. While it's very difficult to give a proper *definition* of turbulence because of the variety of situations in which it can appear, it's possible to identify some features that are common to all turbulent flows:

- **Randomness:** turbulent flows are a typical example of *deterministic chaos*. While the aforementioned equations are almost impossible to solve due to their strong non-linearity, they are still *deterministic*, i.e. certain initial conditions always lead to the same solution of the motion field. But, at high Reynolds number, it's almost impossible to determine the exact initial conditions; moreover, when the flow is turbulent, the solution of the equation is strongly dependent on the initial conditions, i.e. small changes in them lead to great variability of the solution;

- **Diffusivity:** turbulent processes usually involve great rates of mass, momentum and

heat transfer (due to the diffusive term present in the quantity of motion equation);

- **Large Reynolds numbers:** turbulent processes come to existence when inertial forces are much greater than viscous forces; this is due to the fact that inertial forces are responsible for the system instability (so, given a perturbation, the system either diverges or stabilizes in a completely different equilibrium condition), while viscous forces tend to damp perturbations. So, since the Reynolds number is the ratio between inertial forces and viscous forces, it's only natural that when it reaches a certain value (depending on the specific experimental situation) there is transition from laminar to turbulent flow, and the nonlinear convective term in the NS equation becomes predominant;

- **Three-dimensional and rotational:** the flow is never irrotational, i.e. it shows non-null and non-costant vorticity (vorticity is the infinitesimal rotation of the velocity field, $\vec{\omega} = \nabla \times \vec{v}$); in particular, turbulent flows are characterized by an high, fluctuating vorticity. These fluctuations (with large magnitude) could never maintain themselves if the motion field wasn't 3D;

- **Dissipation:** even if at first they are negligible, viscous shear stresses always convert kinetic energy in thermal energy, so at some point in time (if there is no external energy to maintain the flow) turbulence will eventually decay.

So, given that it is impossible to solve the equations analytically, some statistical approaches have been proposed, based on the average values of the quantities involved. The first step is considering every variable as the sum of its mean value and the fluctuation: $f(\vec{x}, t) = \langle f \rangle + f'(\vec{x}, t)$; thus, we can rewrite the NS equations for the average values, obtaining the **Reynolds-averaged Navier-Stokes Equations** (RANS)[1].

It's important to notice that the dynamics of a turbulent flow are deeply influenced by the fact that there is a variety of 'scales' at which they happen, i.e. the vorticose structures (called *eddies*) can be more or less wide.

This leads to another approach called **LES** (*Large Eddy Simulation)*, in which the calculation is simplified by excluding the small scale solutions (that are the most computationally expensive) with a filter (as we will se later, small eddies are associated to high frequencies, so usually a low-pass filter is used). Still, the most accurate method is **DNS** *(Direct Numerical Simulation)*, in which the NS equations are solved directly without a

turbulence model. Obviously this method is the most expensive (computational cost grows with Re^3).

The interaction between big and small eddies is the real responsible of the overall behaviour of the flow. The statistical approach for characterizing these scales was developed by the Russian studious Andrey Kolmogorov, whose theory is briefly summarized below.

An eddy is characterized by a dimensional scale *l*, a velocity scale *u(l)* and subsequently a temporal scale $\tau(l) \coloneqq l/u(l)$. Usually, bigger eddies have a characteristic length $l_o$that is comparable to the dimension of the phenomenon *L*. Their Reynolds number $Re_0 = \dfrac{u(l_0)l_0}{\nu}$ is quite large, i.e. the effect of viscosity is negligible. According to Kolmogorov (and Richardson, who first theorized this idea), bigger eddies (that contain most of the energy of the flow) are unstable, so they tend to decay transferring their energy to smaller ones. This continues to happen in smaller and smaller eddies, until the Reynolds number is so low that viscosity has a relevant effect and kinetic energy is dissipated into thermal energy. This process is known as **energy cascade**. Considering that the eddies that start the cascade have energy proportional to $u_0^2$and temporal scale $\tau_0 = \frac{u_0}{l_0}$, the rate of energy dissipation is $\varepsilon = \frac{u_0^2}{\tau_o} = \frac{u_0^3}{l_0}$ (so it does not depend on kinematic viscosity). In order to link large scales to smaller ones, Kolmogorov introduced three hypotheses.

1.1 Kolmogorov hypotheses[6]

- **First Kolmogorov hypothesis (local isotropy)**
  For Reynolds numbers large enough, small scale ($l \ll l_0$) turbulent motions are statistically isotropic (i.e. they have no preferred direction)
  Kolmogorov theorized that when large scale eddies decay, information about their geometry and their direction is lost. As a direct consequence, for Reynolds numbers adequately large, small scale motions are **universal**, meaning that they are similar in every high-Re flow. The explanation of this fact resides in the assumption that the statistic properties of the small scale dynamics depend uniquely on the energy dissipation rate ε and the kinematic viscosity ν.

- **Second Kolmogorov hypothesis (first hypothesis of similarity)**
  For Reynolds numbers large enough, small scale ($l < l_{EI}$) statistics are universally determined by ε and ν

$l_{EI} \approx {}^1\!/_6 l_o$is the scale at which the universality principle is valid. For every $l < l_{EI}$ (universal equilibrium range), by dimensional analysis the following scales (for dimension, time and velocity) are defined (**Kolmogorov microscales**):

$$\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{1/4}$$

$$\tau_\eta = \left(\frac{\nu}{\varepsilon}\right)^{1/2}$$

$$u_\eta = (\nu\varepsilon)^{1/4}$$

Kolmogorov showed that if one normalizes lengths and velocities with the scales defined above, the motion field is self-similar, i.e. in every point and for every high-Re flow the motion at small scales is statistically identical if made non-dimensional with Kolmogorov scales. Now that the the properties of large scale and small scale eddies are defined, there are only intermediate eddies left to deal with, i.e. all the structures large enough to not be influenced by viscosity, but small enough to be included in the universal equilibrium range ($\eta \ll l < l_{EI} < l_0$)

- **Third Kolmogorov hypothesis (second hypothesis of similarity)**
  For Reynolds numbers large enough, eddies with $l_{DI} \ll l \ll l_{EI}$ are influenced only by ε and not ν
  $l_{DI} \approx 60\eta$ is the scale that divides the universal equilibrium range in the **inertial range** and **dissipation range**. In the inertial range, viscosity does not have a relevant effect yet, so energy transfer is only influenced by inertial effects.

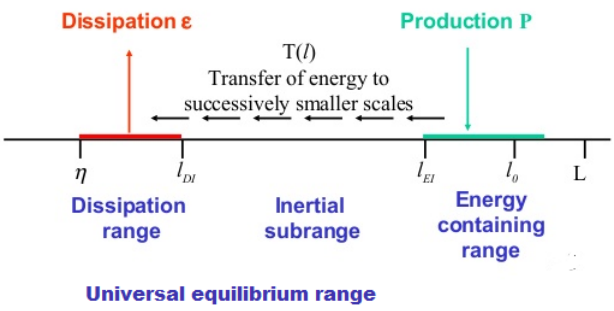


Figure 1.1: Graphic representation of the scales introduced

Chapter 2

Complex network theory

In the last decades, the concept of analysing dynamic time series by converting them into a complex network system has been introduced. In particular, it's been proved that the derived networks inherit some of the features of the parent time series. Before speaking about the transformation algorithms, it's useful to give the reader some general information about graph theory.

A network (meaning a *graph*) is basically a set of **nodes** (or *vertices,* or *points*) connected by **edges** (or *lines*). Their easy structure makes them suitable to model any kind of system in which there are a certain number of interacting entities. Graph theory has been strongly developed in the last century, and its range of applications is virtually countless; infact, while at the beginning graphs were (successfully) used to solve optimization problems (theoretical computer science, game theory, city planning), they were later discovered to be an extremely powerful tool to study biological systems, social groups and neural networks. Suffice it to say that for example Google's PageRank algorithm to index web pages is based on graph theory, combined with statistic tools. In this context, a *complex network* is a graph with non-trivial features, derived from a real system and usually involving thousands or millions of nodes. In the next chapter a brief overview of definitions and metrics of a complex network will be given.

2.1 Main definitions and notations

In general there are four types of graphs: *weighted graphs (directed), unweighted graphs (directed), undirected weighted graphs* (simply called weighted graphs) and *undirected*

unweighted graphs; in particular, the last three are special cases of the weighted directed graph.

From a mathematical point of view, a directed weighted graph *G(N, K) = (V, L)* is defined by a set *V* of *N* nodes and a set *L* of *K* vertices. Each node can be identified by an integer index *i = 1, 2, ..., N* and each edge is identified by a couple *(i, j)*, in which *i* and *j* are the endpoints of the link, and a real non-negative number $\omega(i, j)$ that is the *weight* of the link. A common assumption is that there are no loops (i.e. there are no links between a node and itself) and no multiple connections (i.e. each pair of nodes has no more than one link). A graph that shows one of these two features is called *multigraph* or degenerate graph. In matricial notation, a graph can be completely represented by a *weight matrix* W, in which every entry $w_{ij} = \omega(i, j)$. It's important to notice that in the case of directed graphs the relation $w_{ij} = w_{ji}$is not verified, meaning there's no correlation among the edges entering and exiting a node.

An *unweighted graph* is a graph in which the edges have no weight, so all the links are equipollent. Unweighted graphs are represented by the **adjacency matrix** A, in which the entries $a_{ij}$are defined as following:

$$a_{ij} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are linked} \\ 0 & \text{otherwise} \end{cases}$$

In the case of an *undirected graph* the edges have no direction, i.e. $a_{ij} = a_{ji}$. In these graphs, two nodes *i* and *j* are *neighbours* if $a_{ij} \neq 0$. The *neighbourhood* of a vertex *i v(i)* is the set of nodes adjacent to *i*, i.e. all the nodes with which *i* has a link. The *distance* between two nodes *d(i,j)* is the shortest walk between the two nodes that are connected (meaning there is a walk that goes from *i* to *j*). A *trail* is a walk in which no edge is repeated, while a *path* is a walk in which no node is visited more than once. In this work we will only deal with undirected unweighted graphs, so all the following definitions will be given implicitly assuming that we are considering such a graph.

2.2 Metric properties

There is a number of parameters that can express the topological features of a graph. We will be focusing on the following[8]:

- **Degree**
  The degree (*connectivity*) k_i of a node *i* is defined as the number of edges incident in that node; in terms of adjacency matrix it can be calculated as follows:

$$k_i = \sum_{j \in \aleph} a_{ij}$$

  $\aleph$ is the *neighbourhood* of node *i,* i.e. the set of nodes with which *i* shares an edge. One of the most basic characterizations is the *degree distribution P(k)*, defined as the probability that a randomly chosen node has degree *k* or, similarly, as the fraction of nodes in the graph having degree *k.* It is obviously possible to introduce the mean degree of the graph $\langle k \rangle = \sum_k kP(k) = \frac{1}{N}\sum_{i,j=1}^{N} a_{ij}$. The nodes with high degree are usually called **hubs**.

- **Shortest path length**
  It is useful to have an idea of how 'compact' a network is, i.e. what's the distance between two nodes in the graph. Moreover, it could be easier to identify the most important nodes of the network as the ones most closely connected to the others. A matrix $\mathcal{D}$ is introduced, where every entry d_{ij} is the shortest path between node *i* and node *j*. The maximum value of d_{ij} is called **diameter** of the graph. The most common measure of the typical separation between two nodes is the *average path length*, also known as *characteristic path length*, is the mean of geodesics (shortest paths) over all the couples of nodes in the graph:

$$L = \frac{1}{N(N-1)} \sum_{i,j \in N, i \neq j} d_{ij}$$

  This definition is inconsistent if there are *disconnected* components in the graph (i.e. $\exists i, j : d_{ij} = \infty$), but it will suffice for the purposes of this work.

- **Clustering coefficient**
  Clustering is a typical property of social networks, where to two people who have a friend in common are likely to know each other. Clustering (also known as *transitivity*) can be measured with various coefficients; we will focus on the **clustering**

coefficient C. Firstly, a *local clustering coefficient* c_i is defined; this coefficient expresses how likely is $a_{jm} = 1$, where *m* and *j* are two neighbours of node *i*. This is done by counting the actual number of edges among the nodes in the neighbourhood of node *i* and dividing it by the maximum number of links $k_i(k_i - 1)/2$:

$$c_i = \frac{2e_i}{k_i(k_i - 1)} = \frac{2\sum_{j,m} a_{ji}a_{mi}a_{jm}}{k_i(k_i - 1)}$$

The global clustering C is the average of the c_i over all the nodes in the graph:

$$C = \langle c \rangle = \frac{1}{N}\sum_{i \in N} c_i$$

- **Modularity**
  The modularity of a network is a measure of how much the network is divided in communities (clusters). A high value of modularity expresses the fact that the network is strongly divided into groups; letting $s_i = 1$if node *i* belongs to group 1 and $s_i = -1$if it belongs to group 2, modularity *Q* is defined as:

$$Q = \frac{1}{4m}\sum_{ij}\left(A_{ij} - \frac{k_i \cdot k_j}{2m}\right)s_i s_j$$

  where $m = {}^1\!/_2 \sum_i k_i$ is the total number of edges in the graph and $k_i k_j/2m$ is the expected number of edges between *i* and *j* if the edges were placed randomly. Modularity is useful because nodes belonging to the same group are likely to share the same properties; above all, there are a lot of connections inside the community, but links become less dense among the different clusters, so the information transmission rate is much higher inside a single group than between a group and another.

Chapter 3

The visibility algorithm

Now that we have a little background information, we can deal with the generation of the graph. As mentioned before, the idea is to transform a time series data into a unweighted undirected graph, and study its topology in order to gather some informations about the underlying series. Various algorithms have been proposed to achieve this objective (for example the phase space method, introduced by Xu et al.[5]). In this work we will use the *visibility algorithm*, proposed by Lacasa et al.[4] because of its simplicity.

Let $x(t_i) = x_1, ..., x_N$be the values of a time series of *N* measurements and imagine to plot them in a vertical bar diagram. In Figure(3.1) we have an example of a $|sin(t)|$plotted between 0 and π discretized in 16 equidistant points. In this diagram, we link every bar with all the bars that are visible from the top of the chosen one with a straight line. Thus, we can obtain a graph in which every point is a node and a link exists if two nodes are visible from one another.

Formally, we can enunciate a *visibility criteria*: two arbitrary data points (t_a, y_a) and (t_b, y_b) are visible (and thus are connected in the visibility graph) if for every data (t_c, y_c) between them the following relation is verified:

$$y_c < (y_a - y_b)\frac{t_b - t_c}{t_b - t_a}$$

The graph obtained with this algorithm shows some interesting features; in particular, it is:

- Connected: each node sees at least its neighbours (left and right);

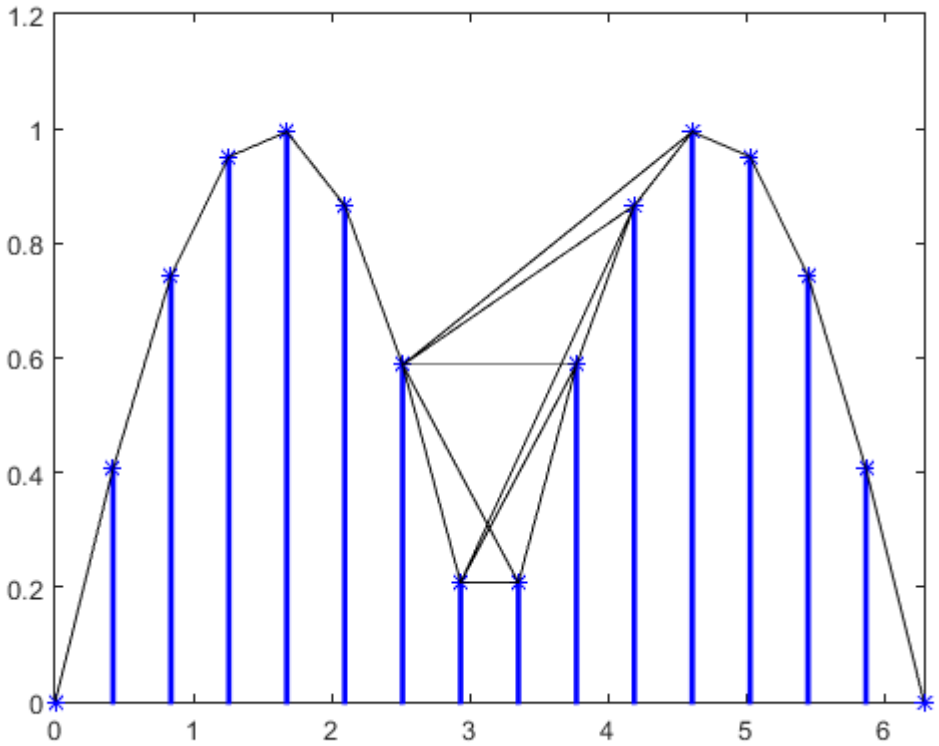


Figure 3.1: Visibility algorithm

- Undirected: the existence of a link between node a and b implies a link between b and a (although the algorithm could be easily built so that it would distinguish ingoing and outgoing connectivity);
- Unweighted: all the links are equivalent, so the algorithm returns an unweighted network;
- Invariant under affine transformation: the visibility graph doesn't change if the time series is rescaled horizontally or vertical or if it is translated along the x or y axis.

The constructed graph inherits some properties of the underlying time series: periodic series produce regular graphs, as well as random series convert into random graphs. In the present work, a MATLAB script has been devised; this script, given a time series in form of a vector, returns the adjacency matrix relative to its visibility graph and some representative

metrics.

Chapter 4

Results of the simulation

4.1 The JHTDB database

As mentioned before, in this work a turbulent flow in a channel is studied. In particular, we will investigate the topology of two visibility graphs obtained by time series measured in different regions of the turbulent flow: we will consider a point in the buffer layer and a point in external region; in order to have a wider perspective on the various regions of the flow, we also considered three additional points (wall region, channel axis and one in between the two aforementioned). As mentioned before, the points are taken along the wall (because it is the inhomogeneous direction), so they share the same x and z coordinates (they were taken at about the center of the channel).

The data has been retrieved from the John Hopkin's Turbulence Database (JHTDB), an open-access freely available online (http://turbulence.pha.jhu.edu/). Despite the impossibility to solve deterministically in a closed form the NS equations, thanks to the modern development of computational power of electronic devices many numeric resolution techniques have been effectively used. The data in this database is obtained via a DNS (Direct Numerical Simulation), meaning that the NS equations are solved directly without any turbulence model. DNS is usually very expensive from a computational point of view, and it is practically inapplicable with flows at high Reynolds number (the computational cost grows with Re^3). Still, it is used with simple configurations such as isotropic homogeneous turbulence and channel flow; also, it is applied to study turbulences for research purposes, because the conditions needed for theoretical studies usually can't be recreated experimentally. The database can be accessed via a web interface (for data cutout) or dir-

ectly via a program, using APIs developed for many programming languages. Given that we used only two points, we downloaded the data using the web interface in .h5 format and extracted it with a MATLAB script.

4.2 Characteristics of the flow

The simulations parameters were the following:

- Domain dimensions: $L_x \times L_y \times L_z = 8\pi h \times 2h \times 3\pi h$, where *h* is half the channel height;
- Grid: $N_x \times N_y \times N_z = 2048 \times 512 \times 1536$
- Timestep: $\delta t = 0.0065$;
- Time stored: $t = [0.0, 25.9935]$

So, we have 2048 x 512 x 1536 points in the gridspace, each one sampled at 4000 timesteps. The flow had the following characteristics:

- Viscosity: $\nu = 5 \cdot 10^{-5}$;
- Mean pressure gradient: $dP/dx = 0.0025$;
- Bulk velocity: $\bar{U} = 0.99994$;
- Reynolds number over full channel height and bulk velocity: $Re = \dfrac{2h\bar{U}}{\nu} = 3.9998 \cdot 10^4$;
- Friction velocity: $u_\tau = 4.9968 \cdot 10^{-2}$;
- Viscous length scale: $\delta_v = \frac{\nu}{u_\tau} = 1.0006 \cdot 10^{-3}$;

All values are given in non-dimensional units.

We report below a table summarizing the various regions of a turbulent flow near wall:

In order to examine network topology differences between two regions of the flow, we took a point in the buffer layer ($y_1 = 20\delta_v$) and a point in the external region ($y_2 = 70\delta_v$).

Region	Position	
Viscous sublayer	$y < 5\delta_v$	Viscous friction is predominant over turbulent friction
Buffer layer	$5\delta_v < y < 30\delta_v$	Transition between viscous sublayer and log law
Logarithmic law region	$30\delta_v < y < 0.3h$	Log law is valid
Wall viscous region	$y < 50\delta_v$	Viscosity contribution to total friction is relevant
External region	$y > 50\delta_v$	Viscosity direct effect is negligible

Table 4.1: Regions of a turbulent flow near wall

4.3 The networks

As mentioned before, two points were considered. In grid coordinates, they were $P_1 = (1024, 491, 800)$ (buffer layer) and $P_2 = (1024, 500, 800)$ (external region), corresponding to $(12.5664, 9.9204 \cdot 10^{-1}, 4.9087)$ and $(12.5664, 9.9752 \cdot 10^{-1}, 4.9087)$; the coordinates are made dimensionless with the half-channel height h. As characteristic parameter, the energy of the flow was chosen, because it is a scalar (hence easier to work with) and gives a general overview of the features of the motion field. In order to determine it, the velocity field (u, v, w) in both points was downloaded, the we calculated the energy as:

$$E(x, y, z) = \frac{u^2}{2} + \frac{v^2}{2} + \frac{w^2}{2}$$

The energy trend in time is the following:

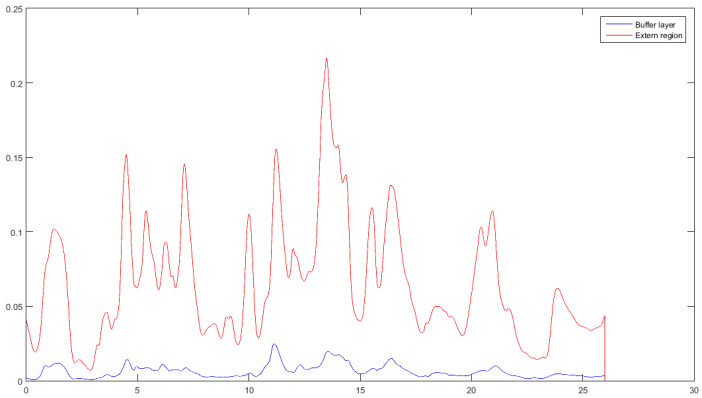


Figure 4.1: Energy over time

This way, we can use the visibility algorithm to build the two complex networks:[1]

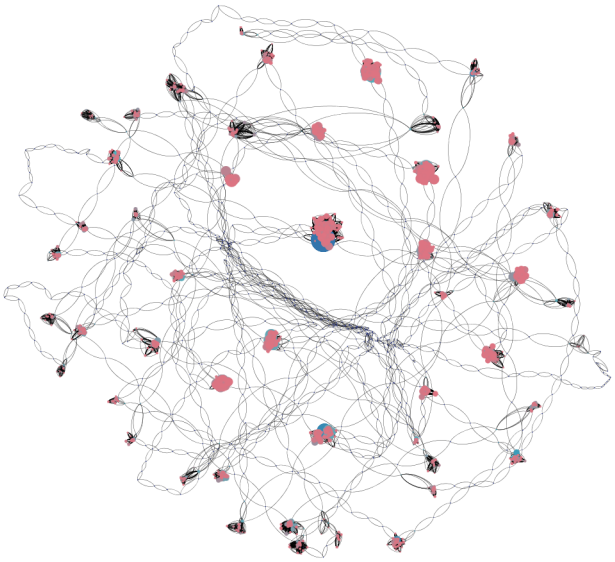


Figure 4.2: Network of the buffer layer point

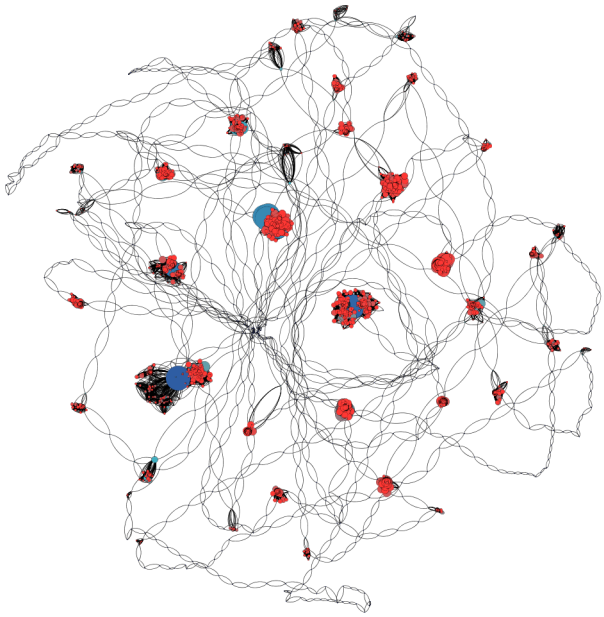


Figure 4.3: Network of the external region point

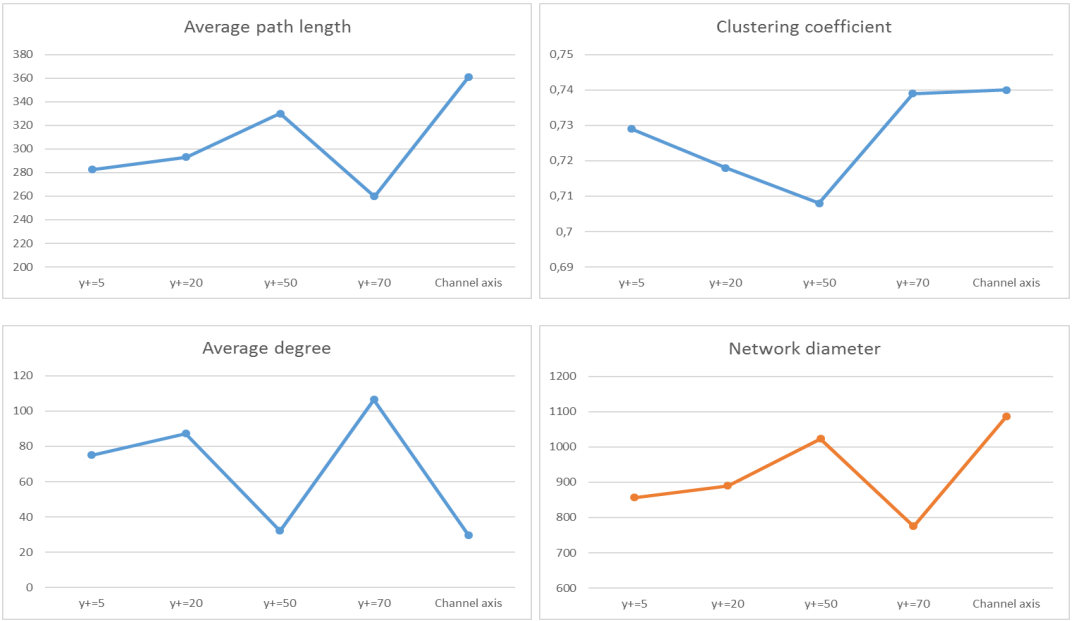
[1]The images were generated using Gephi software https://gephi.org

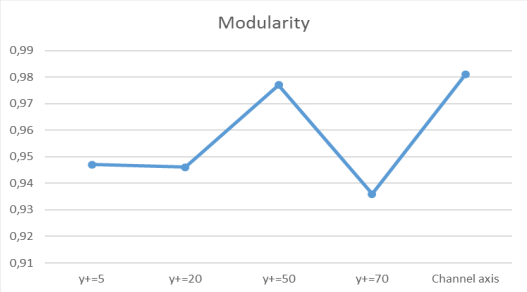
In both graphs, the size of the nodes is proportional to their degree, while their color depends on their clustering coefficient, with a palette that goes from blue to red.We also reported network statistics for two "extreme" situations, in order to have a comparison scale: a near-wall point ($y^+ = 5$) and a point in the channel axis, both with the same x and z coordinates. In order to show the transition an additional point was considered in between the two coordinates considered; this point is located at $y^+ = 50$. Below there are some general properties of the graphs:

	$y^+ = 5$	BL ($y^+ = 20$)	$y^+ = 50$	ER ($y^+ = 70$)	Channel axis
Average degree	75.165	87.313	32.091	106.553	29.679
Average path length	282.621	293.064	329.868	259.693	360.782
Network diameter	857	890	1023	775	1086
Clustering coefficient	0.729	0.718	0.708	0.739	0.7
Modularity	0.947	0.946	0.977	0.936	0.981

Table 4.2: Metrics of the two networks

Firstly we report below the trends of the various metrics among the different points:





4.4 Discussion of the results

It's immediate noticing that the two points closest to the channel wall are very similar, i.e. they show almost identical topological parameters; as we move towards the center of the channel, the complex networks' metrics begin to have a different evolution pattern. Of course these variations are directly linked to the underlying dynamic system.

Let us consider again the two points that are closest to the wall. They show low values for diameter and average path length, meaning that the networks are 'dense', so their nodes are connected pretty tightly. As mentioned in Chapter 1, a fully-fledged turbulence contains smaller and bigger eddies, with bigger eddies being also the most energetic. Having said that, one can infer that near the channel walls viscosity has a major effect, so only small, short-living eddies survive (bigger ones are disrupted); then, having a small network, with shorter-memory connections is coherent with the physical system underneath it. From this point of view, it's only natural that the network relating to the channel axis has just opposite features: larger diameter, larger average path, because there are larger, more energy-containing structures made up of connections with longer memory.

Now let us focus on the point in the external region, which is quite a special point,because it's on the boundary between the viscosity-influenced region and the region out of the log law space. This point shows particularly high degree and clustering coefficient, and at the opposite very low modularity. Once again, this reflects the behaviour of the flow: the way the visibility graph is constructed, points with higher degree are usually points with high energy surrounded by less energetic points; in this context, it makes sense that a point in the external region has a really high degree, because most of the points in the log law will have lower velocity (thus lower energy) and so will be 'visible' from it. This theory is corroborated by the fact that a point near the channel axis is likely to have many neighbours

with comparable energy, so it will probably have a significantly lower degree, as it actually happens in this case. As for the modularity, from a network point of view it's dependent on the distribution in communities. Generally speaking, high modularity means strong division in communities; in particular, the higher the modularity, the fewer the communities. So, considering that in the external region the effect of viscosity begins to be negligible, there is a stronger influence of larger eddies; this leads to a transition situation, because there is a chaotic interaction between large and small structures. Modularity drops drastically, because there is no time to develop an organized system with few communities and the nodes are linked very similarly. As opposed to this, the channel axis network shows a much larger modularity: the flow is organized in large eddies, with longer memory, and graph represents it exactly.

Let us now focus on the spatial variation of the clustering coefficient. In social networks, the clustering coefficient measures the probability of the 'friends' of a node i being acquaintances. In this case, the points directly influenced by viscosity see, as said before, a flow animated by small, short-living structures. These eddies provoke a strong, deep perturbation of the system, which is responsible for a faster loss of memory of the connections between the nodes; because of that, we have low clustering coefficient, Instead, for the point in the external region it's much higher, because the influence of larger eddies becomes stronger.

So then, it is possible to infer the following:

- The point in the buffer layer is highly influenced by viscosity; the network parameters show clearly that its evolution is dictated by short memory structures, whose continuous decaying exerts a non-negligible perturbation on the system's behaviour; the nodes of the graph are quite independent and the overall situation doesn't really depend on the previous instants.

- As for the external region point, the situation is a bit less obvious: the general behaviour cannot be brought back to the exclusive action of small eddies, nor bigger vortices, but rather to the combined influence of both; infact, for all of the parameters analysed, there is a sudden trend inversion, with properties significantly different compared to the points closer to the wall. Still, the properties are also quite different also from the network of the channel axis, so the overall dynamic is pretty unique and due to the interaction between big and small vortices.

Conclusions

The analysis of the networks provided results that are coherent with the theoretical model illustrated in Chapter 1. Infact, we demonstrated that the regions closest to the wall are dominated by small eddies (as predicted by Kolmogorov's theory), while the region around channel axis is influenced by larger, more energetic vortices. The point in the external region turned out to be a very interesting choice because of the peculiar properties it possesses: the interaction between the different phenomenological scales generates a particular behaviour, quite different from the two beforementioned. Network analysis demonstrated that short-living eddies generate smaller, less connected graphs, because there is no time to develop a complex structure with many connections. On the contrary, points in the range of longer living structures have bigger networks, less dense and more clustered.

By examining a system that's been thoroughly analysed in countless other works, it is possible to verify the trustworthiness of the complex network approach to this kind of problems. In particular, in this case the interpretation of the results was coherent with the mathematical model commonly used to deal with turbulence problems: it is worth noticing that unfortunately the visibility algorithm (at least in the way it was implemented in this work) doesn't inherit the absolute "scale" of the problem, but only shows the differences between networks; for example, in this situation we couldn't tell *a priori* which points were more energetic, the difference was inferred only by studying the external region point. Basically, two completely different system (even periodic series with completely different values but same period) could end up having the same visibility graph, if they evolve in time with the same pattern (as mentioned in Chapter 3, visibility graphs are invariant under affine transformations).

So, with a relatively small effort, we were able to gain a variety of information about the various points in the flow. The visibility algorithm proved really effective in differential

analysis, because interpreting the differences between the networks lead us to hypothesize a physical situation coherent with Kolomogorov's turbulence models. Thus, it is possible to develop a systematic method to operate a preliminary characterization of the regions in turbulent flows (but it could be easily extended to other phenomena), and with a fast analysis gain an overall knowledge of the physical system.

In particular, the visibility algorithm is fast and easy to implement, and constructs a graph very easy to interpret: infact, given the way it is built, it immediately shows how every node (which is, after all, a measurement in a given instant of time) is connected to all the others, thus providing a straightforward insight on the dynamic changes of the system and to which extent the configuration at a certain instant of time depends on the situation at the previous instants. Finally, complex network has been confirmed to be a very promising technique to perform a characterization of turbulent flows, and the visibility algorithm proved to be a good option to generate a graph to study.

Bibliography

[1] Renzo Arina, *Fondamenti di aerodinamica*, Levrotto & Bella, 2016.

[2] Panos Papanicolau A. Liakopoulos Avraam Charakopoulos, Theodoros E. Karasidis, *The application of complex network time series analysis in turbulent heated jets*, Chaos: An interdisciplinary journal of nonlinear sciences (2014).

[3] John Hopkins Turbulence Database, *http://turbulence.pha.jhu.edu/*.

[4] Lacasa et al., *From time series to complex networks: The visibility graph*, PNAS **105** (2008), no. 13.

[5] Xu et. al, *Superfamily phenomena and motifs of networks induced from time series*, PNAS (2008), 105.

[6] Andrey Nikolaevich Kolmogorov, *The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers*, Proceedings of the Royal Society A (1991).

[7] Stephen B. Pope, *Turbulent flows*, Cornell University, 2000.

[8] Y. Moreno M. Chavez D.-U. Hwang S. Boccaletti, V. Latora, *Complex networks: structure and dynamics*, Physics Reports **424** (2006), 180–184.

[9] Gephi software, *https://gephi.org*.

Appendix

Below there is the code used to generate the adjacency matrix from the time series and to calculate its properties.

```
%GENERATE ADJACENCY MATRIX FROM TIME-SERIES DATA
%AND CALCULATE ITS METRICS
clear all
clc
t = 0:0.0065:25.9935;
load('energy_fields.mat');
N = length(t);
series = energy_delta_50;
%Calculate adjacency matrix
A = ones(N);
for i = 1:N
    A(i, i) = 0; %Elements on the main diagonal are null by defin
end
for i = 1:N
    c = 0;
    for j = (i+2):N
        if (c~=0)
            A(i, j) = 0;
            continue;
        end
        for k = (i+1):j %Visibility condition
           if(series(k)>(series(j)+(series(i)-series(j))*(j-k)/(
```

```
                c=1;
            end
            if (c~=0)
                A(i, j)=0;
                            break;
            end
        end
    end
end
for i = 1:N %The adjacency matrix is symmetric
    for j = (i+2):N
        A(j, i) = A(i, j);
    end
end
%Floyd-Warshall algorithm for shortest paths
D = repmat(inf, N, N); %I start with every vertex disconnected
for i = 1:N
    D(i, i) = 0;
%The distance between a vertex and itself is 0 end
for i = 1:N
    for j=1:N
        if (A(i,j) == 1)
            D(i,j)=1; %Shortest path between neighbours = 1
        end
    end
end
for k = 1:N
    for i = 1:N
        for j = 1:N
            if D(i,j) > D(i,k) + D(k, j)
                D(i,j) = D(i,k) + D(k,j);
            end
        end
```

```
    end
end
K = zeros(1, N); %Connectivity vector
for i = 1:N
    for j = 1:N
        K(i) = K(i) + A(i,j);
    end
end
%Average distance and diameter of the graph
diameter = 0;
sum = 0;
for i = 1:N
    for j = 1:N
        sum = sum + D(i,j);
        if (D(i,j) > diameter)
            diameter = D(i,j);
        end
    end
end
average_distance = sum / (N*(N-1));
% disp('Distanza media ');
% disp(average_distance);
% disp('Diametro ');
% disp(diameter);
% disp(K);
fid = fopen('graph_adjacency.dat', 'wt');
for i = 1:size(A, 1)
    fprintf(fid, '%d ', A(i, :));
    fprintf(fid, '\n');
end
fclose(fid);
fid = fopen('graph__metrics.dat', 'wt');
fprintf(fid, 'Diameter: %g\n', diameter);
```

```
fprintf(fid, 'Average path length: %g\n', average_distance);
fprintf(fid, 'Connectivity vector: \n');
for i = 1:N
    fprintf(fid, '%d ', K(i));
end
```

The file "energy-fields.mat" is obtained by extracting the velocity field from the .h5 file downloaded by the JHTDB and calculating the kinetic energy:

```
clear all
clc

velocity_buffer_layer = zeros(3, 4000);
velocity_external_region = zeros(3, 4000);
external_region = hdf5info('channel_70_delta_punto_491.h5');
buffer_layer = hdf5info('channel_20_delta_punto_500.h5');
energy_buffer_layer = zeros(1, 4000);
energy_external_region = zeros(1, 4000);
energy_delta_5 = zeros(1, 4000);
delta_5_file = hdf5info('channel_5_delta_punto_506.h5');
velocity_delta_5 = zeros(3, 4000);
axis_file = hdf5info('channel_axis_punto_257.h5');
energy_axis = zeros(1, 4000);
velocity_axis = zeros(3, 4000);
delta_50_file = hdf5info('channel_50_delta_punto_460.h5');
velocity_delta_50 = zeros(3, 4000);
energy_delta_50 = zeros(1, 4000);
for i = 1:3999
     a1 = hdf5read(buffer_layer.GroupHierarchy.Datasets(i+4));
     a2 = hdf5read(external_region.GroupHierarchy.Datasets(i+4));
     velocity_buffer_layer(:, i) = a1;
     velocity_external_region(:, i) = a2;
     a2 = hdf5read(delta_5_file.GroupHierarchy.Datasets(i+4));
```

```
     velocity_delta_5(:, i) = a2;
     a1 = hdf5read(axis_file.GroupHierarchy.Datasets(i+4));
     velocity_axis(:, i) = a1;
     a1 = hdf5read(delta_50_file.GroupHierarchy.Datasets(i+4));
     velocity_delta_50(:, i) = a1;
end

for i = 1:4000
     energy_buffer_layer(i) = (velocity_buffer_layer(1, i)^2 \
+ velocity_buffer_layer(2, i)^2 + velocity_buffer_layer(3, i)^2)/
     energy_external_region(i) = (velocity_external_region(1, i)^
+ velocity_external_region(2, i)^2 + velocity_external_region(3,
     energy_delta_5(i) = (velocity_delta_5(1, i)^2 \
+ velocity_delta_5(2, i)^2 + velocity_delta_5(3, i)^2)/2;
    energy_delta_50(i) = (velocity_delta_50(1, i)^2 \
+ velocity_delta_50(2, i)^2 + velocity_delta_50(3, i)^2)/2;
        energy_axis(i) = (velocity_axis(1, i)^2 \
+ velocity_axis(2, i)^2 + velocity_axis(3, i)^2)/2;
end

save energy_fields.mat energy_axis energy_delta_5\
energy_delta_50 energy_buffer_layer energy_external_region;
```