UTRECHT UNIVERSITY

DOCTORAL THESIS

# Intensification of tropical cyclones by gradient wind adjustment due to diabatic heating

*Author:*
Jasper DE JONG

*Supervisor:*
Dr. Aarnout VAN DELDEN

*Second examiner:*
Dr. Michiel BAATSEN

*A thesis submitted in fulfillment of the requirements for the degree of Master of Science*

*in the*

Atmospheric Dynamics
Institute for Marine and Atmospheric research Utrecht (IMAU)

March 4, 2020

UTRECHT UNIVERSITY

# *Abstract*

Graduate School of Natural Sciences
Institute for Marine and Atmospheric research Utrecht (IMAU)

Master of Science

**Intensification of tropical cyclones by gradient wind adjustment due to diabatic heating**

by Jasper DE JONG

In this research project the intensification of tropical cyclones is interpreted as a process of gradient wind adjustment, much like the process of geostrophic adjustment, which was investigated first by Rossby in the 1930's. The project consists of two aspects: a modelling aspect and a diagnostic aspect.

The modelling aspect is concerned with investigating the process of gradient wind adjustment in Ooyama's (1969) two-layer simplified model of a tropical cyclone. Latent heat release in clouds in the center of the tropical cyclone is prescribed by a "diabatic mass flux" from the lower main layer to the upper main layer. This mass flux is an analogy to cross-isentropic flow (diabatic heating) in the real atmosphere. The diabatic mass flux in the model, which is proportional to convergence in the (Ekman-) boundary layer, changes the depth of the bottom and top model layer, which alters the potential vorticity distribution, excites gravity-inertia waves and an adjustment to gradient wind balance. The diabatic mass flux is prescribed by three parameters: a radius of maximum mass flux, $r_0$, a width, $a$, and a maximum intensity, $Q_0$.

The model results show that the ratio of kinetic energy in the final balanced state, relative to the input of potential energy by diabatic heating, is optimal (a maximum) for diabatic mass flux distributions with both $r_0$ and $a$ in the order of the local Rossby radius of deformation in the centre of the cyclone. The central deepening rate seems to vary proportionally to $a$ divided by the local Rossby radius.

The diagnostic aspect of this project is concerned with determining the diabatic heating distribution in a high-resolution simulation of hurricane *Irma* (2017) with HARMONIE, the limited area weather forecast model of KNMI. The diabatic heating is found to be positive in regions of latent heat release, consistent with the precipitation, such as in the eyewall, and negative in regions of descending air, such as inside the eye. The growth of the cyclone, in terms of the total kinetic energy, although just being one example, seems to be in line with gradient wind adjustment theory.

# *Acknowledgements*

During the course of the current research, I have received help from others for which I am very grateful. Firstly, I would like to thank my supervisor dr. Aarnout van Delden for many hours of discussing ideas for making progress in this study, providing relevant intellectual sources and motivating me to think about implications of the found results. Secondly, I would like to thank my girlfriend, Anja, for supporting me through the whole process of doing this research. Last, but not least, I am very thankful for dr. S. Tijm of the KNMI for providing high resolution data of hurricane Irma that has empowered me to make detailed analyses of a realistically simulated tropical cyclone. All above mentioned help has greatly contributed to enhancements in the quality of this project.

# Contents

# List of Figures

# Chapter 1

# Introduction

Tropical cyclones are very interesting phenomena, arguably comprising all subjects of classical fluid dynamics. At the smallest scales, we find air flow around water droplets, coalescence of droplets, phase transitions of water. At larger scales we can look at the organized convective motions, fueling the primary swirling flow onto which inertia-buoyancy waves and vortex Rossby waves can propagate. The coexistence of physical processes at such vastly different scales make it currently impossible to fully simulate a tropical cyclone, and therefore parametrizations of small-scale processes are very important.

Current cyclone modelling research is strongly focused on capturing as many processes as possible, by increasing vertical and horizontal resolution. The aim of the current project is not to follow this approach, but instead focus on an often forgotten theory. Gradient wind adjustment is the process by which a vortex in gradient wind balance responds to perturbations such as by diabatic heating. This adjustment has been studied for large scale circulations but never for tropical cyclones.

By investigating the process of gradient wind adjustment in a tropical cyclone, its intensification might be understood in a very comprehensive way.

Investigation of the consequences of gradient wind adjustment in the simulation of tropical cyclone growth will be performed in the following way. First, the process of gradient wind adjustment is investigated in the simplified 2-layer model as described by Ooyama, 1969. The vertical mass flux $Q^+$ from the lower main layer to the upper main layer will be prescribed by an exponential function of the radius, $r$. The parameters of this vertical mass flux, which is an analogy to cross-isentropic flow due to diabatic heating in the real atmosphere, are the amplitude, $Q_0$, the width, $a$, and the radius of maximum heating, $r_0$. Varying these parameters will provide more insight in the adjustment process of a balanced axisymmetric vortex. Second, high resolution data of hurricane Irma is processed, using the continuity equation in isentropic coordinates, to determine the diabatic heating. These results will check whether the intensification or deepening rate of a tropical cyclone can be understood as a result of gradient wind adjustment.

The above described methodology will hopefully shine a light on the largely unknown possibilities of gradient wind adjustment theory in tropical cyclone growth modelling and lead to improved theoretical understanding of tropical cyclone intensification.

# Chapter 2

# Theory

## 2.1 Tropical cyclones

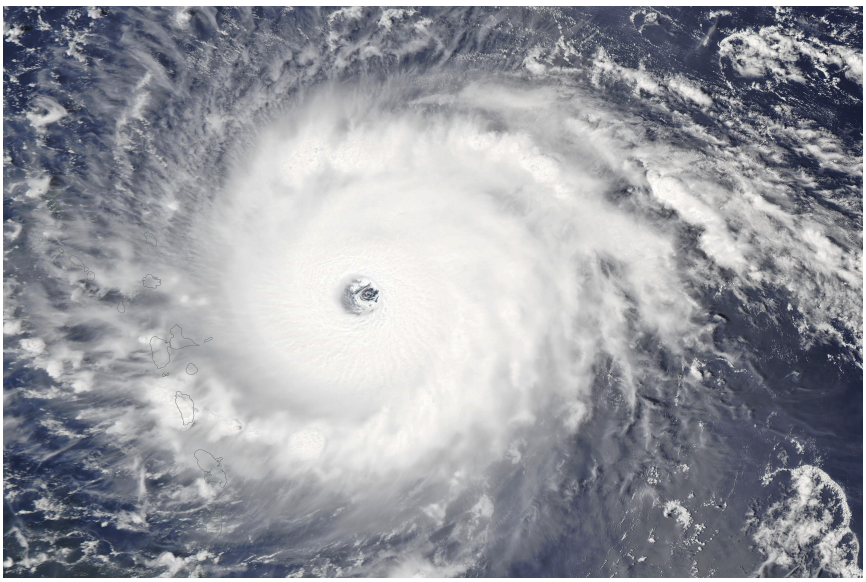

FIGURE 2.1: Moderate Resolution Imaging Spectroradiometer (MODIS) image of hurricane Irma approaching the Caribbean islands on 5 September 2017. Source: National Aeronautics and Space Administration (NASA)

'Tropical cyclone' is the name used to describe a rotating, organized system of clouds and thunderstorms that originates over (sub)tropical waters and has a closed, low-level circulation. In their weakest form they are called tropical depressions. The criterion used by the National Oceanic and Atmospheric Administration (NOAA) to distinguish weak cyclones from strong cyclones is the maximum sustained wind speed, the 1-minute average wind speed 10 meters above the surface. If this quantity is below 62.76 km/h, we speak of a tropical depression. Stronger tropical cyclones that have a maximum sustained wind speed below 119.09 km/h are called tropical storms. In case the maximum sustained wind speed is above 119.09 km/h, the term 'hurricane', 'typhoon' or 'tropical cyclone' is coined to describe the weather phenomenon. In the North Atlantic, central North Pacific, and eastern North Pacific, the term hurricane is used. The same type of disturbance in the Northwest Pacific is called a typhoon. Meanwhile, in the South Pacific and Indian Ocean, the generic term tropical cyclone is used, regardless of the strength of the wind associated with the weather system according to the NOAA.

Any tropical cyclone is characterized by a strong wind in the lower troposphere, swirling in a cyclonic fashion around an area of low atmospheric pressure. Under the right circumstances, a tropical cyclone can sustain itself or even grow in intensity by utilizing the available thermal energy near the surface of warm (sub)tropical water basins. In the Atlantic these circumstances occur most often during the hurricane season, which runs from June 1 to November 30. Different amplification mechanisms that are previously proposed in literature to describe the intensification of tropical cyclones will be discussed in the next section.

The main features of a tropical cyclone are the eye, eyewall and multiple rainbands. Generally low-level air, organized in bands of rain-producing convective updraughts, spirals inwards around the central pressure minimum in a cyclonic fashion. At higher altitude, an outflow from the center is present rotating much weaker, or even anti-cyclonically. In the absolute center of the tropical cyclone a region of descending air is found. This region is called the eye, and is associated with a calm, cloud-free sky. The eye is surrounded by the eyewall, a region of ascending air and the highest horizontal wind speeds found in the whole tropical cyclone. This region is characterized by tall convective clouds, torrential rain and very high wind speeds.

## 2.2 Intensification mechanisms

The general response of a balanced vortex to diabatic heating has been captured by a very popular equation, known as the Sawyer-Eliassen equation. Given a diabatic heat source, it provides a means of calculating the overturning circulation that is necessary in order for the vortex to remain in balance. It does not however, provide a way to determine the amount of diabatic heating based on the dynamical flow properties. As this is important in understanding the intensification of tropical cyclones, we will further elucidate on this topic in the current section. Developments in tropical cyclone intensification research of the last few decades can be divided in four paradigms, three of which are based on axisymmetric models. An extensive overview is given by Montgomery and Smith, 2014. A more condensed description of these paradigms in tropical cyclone research is given in the following subsections.

### 2.2.1 The CISK paradigm

In Charney and Eliassen, 1964 the intensification of a tropical cyclone is explained by the effect of friction in the boundary layer. Around the same time, Ooyama was working on a similar theory, which he published independently (Ooyama, 1964). At the time friction was perceived to merely cause a spin down on an incipient vortex. In their paper, Charney and Eliassen state that the friction performs a dual role, namely the dissipation of kinetic energy and the supply of latent heat to the system by frictional convergence. In their axisymmetric linear model they assume that the rate of latent heat release is proportional to the vertically-integrated convergence of moisture in the troposphere. The diabatic heating due to latent heat releases causes a region of inflow in the lower troposphere below the altitude of maximum heating. This inflow occurs mainly in the boundary layer due to the reduced centrifugal and Coriolis force. The generalized Coriolis force in the region of inflow acts to amplify the tangential velocity. The increased tangential velocity at the top of the boundary layer increases the frictionally induced inflow, and hence the moisture convergence, in the boundary layer. By letting the rate of latent heat release be proportional to the moisture convergence the intensification mechanism is complete. Charney and

Eliassen found positive growth rates in their linear model for sub-synoptic scale disturbances, which is interpreted as a form of macro instability. This instability is later named conditional instability of the second kind (CISK), to distinguish it from the instability that is responsible for the formation of single cumulus clouds. The theory has inspired much subsequent research, and was invoked in many text books. Charney and Eliassen noted that in their linear analysis, they assume a near-saturated boundary layer without considering heat or moisture fluxes at the ocean-air interface. They state the limitations of this assumption for the case of a tropical cyclone in the early formative stage, where the air is far from having reached its equilibrium humidity, as well as for the decay of tropical cyclones when passing over extended land masses.

### 2.2.2 The cooperative intensification paradigm

Although Ooyama had published an article on CISK in 1964, he soon saw the limitations of his work. This valuable insight led him to develop a new theory that he, in a later article (Ooyama, 1982), will call cooperative intensification. The cooperative intensification theory explains tropical cyclone intensification in the following way. In a weak axisymmetric vortex, organized convective activity will take place where the frictionally induced inflow converges. The resulting differential heating causes changes in the pressure field and induces a transverse circulation with the sole purpose of restoring balance between the fields of pressure and motion. If the pseudo-equivalent potential temperature of the boundary layer is large enough for deep convection to occur, the transverse circulation will bring in more absolute angular momentum than that is lost by frictional dissipation in the boundary layer. Then the increasing tangential winds must be balanced by a deepening central surface pressure, which increases the convergence in the boundary layer and closes the cycle. The cooperative intensification theory takes into account the effect of the heat fluxes at the ocean-air interface and the progressive decrease of convective instability due to the development of a warm core. While there was a runaway instability of the linear models based on CISK theory, this is not the case with the cooperative intensification theory. As the cyclone matures, two limiting factors start to play a significant role. As the boundary layer humidity increases, the surface moisture flux decreases. Also, as more latent heat gets released at the condensation level, the local conditional instability decreases. These realistic properties have empowered Ooyama to make the first successful simulation of the life cycle of a tropical cyclone, which is characterized by a formative, deepening and filling stage.

### 2.2.3 The WISHE paradigm

Another paradigm shift occurred after the publication of Emanuel, 1986. The acronym, WISHE, stands for wind-induced surface heat exchange. In Montgomery and Smith, 2014, a description of the feedback mechanism is provided. A concise version of this description is shown here. The typical near-surface wind speed in a tropical cyclone except in the eye decreases with increasing radius. This is typically accompanied by a negative radial gradient of the specific humidity and the equivalent potential temperature $\theta_e$. This negative gradient of $\theta_e$ is transferred to the top of the boundary layer by vertical mixing. Because the air parcels rise due to frictional convergence in the boundary layer, while conserving $\theta_e$, the virtual temperature in the cloudy region also has a negative gradient. Thus in this region there is a warm core. If the

radial gradient of the specific humidity by some disturbance would decrease further, the radial temperature gradient in the cloudy region would decrease as well. Assuming thermal wind balance, the (negative) vertical shear of the tangential wind decreases. At the top of the boundary layer this translates into an increase in the maximum tangential wind speed. This increases the gradient of specific humidity and $\theta_e$ again above the ocean surface. The increased specific humidity does have a diminishing effect on the evaporative moisture flux, however, the accompanied decrease in surface pressure increases the saturation specific humidity which could help in sustaining the moisture disequilibrium. A time dependent extension of the 1986 paper was built in Emanuel, 1997, in order to investigate the intensification process. In this paper he derived an equation for the time evolution of the maximum tangential wind. One weakness of WISHE theory was posed by the lack of a rigorous basis for the formulation of a critical component of this equation, according to Montgomery. A second point of critique was the fact that Emanuel had assumed gradient wind balance in the boundary layer in his 1997 model. In (Emanuel and Rotunno, 2011; Emanuel, 2012), Emanuel does come with a revised theory on WISHE which, among other advances, dispensed with the function that lacked a rigorous basis. However in subsequent research with advanced three-dimensional models, such as Montgomery, Persing, and Smith, 2015, it seems that WISHE is no longer seen as being essential for explaining tropical cyclone intensification in three dimensions.

### 2.2.4 The new rotating-convective updraught paradigm

The above mentioned paradigms all assume an axisymmetric vortex. We know from radar and satellite observations that tropical cyclones have a high degree of asymmetry. Only the strongest mature cyclones are axially symmetric, and this only holds close to the center of the cyclone. Recent advancements using models with high horizontal and vertical resolutions have provided results that show much more resemblance to observed tropical cyclones. One such model is given by Van Sang, Smith, and Montgomery, 2008. It is a non-hydrostatic model containing simplified approximations for physical processes where possible, with a horizontal resolution of 5 km and 24 vertical layers. Nguyen et al. assume the $f$-plane approximation. In this way they assure there is no creation of a vorticity dipole due to variation of planetary vorticity, and thus relative vorticity, as parcels travel to different latitudes. Still in the numerical runs, they find that the vortex develops axial asymmetries. One clear feature of the flow during intensification is the existence of multiple strongly rotating convective updraughts that form during a period of about 2 days after the initialization of deep convection at several locations in the radius of maximum wind. These updraughts rotate around the center of the cyclone, and are observed to have a life time in the order of one hour. By stretching and tilting they create dipoles in the relative vorticity field. The anomalies of positive vorticity, which are stronger than their negative counterparts, move towards the center and become segregated from the negative vorticity anomalies. The relative vorticity of these updraughts is found to be much greater than that of the parent vortex. Eventually, the updraughts merge and undergo axisymmetrization by the parent vortex. During the numerical run of Sang et al. the amount of these updraughts reduces from 12 at the beginning of a period of rapid intensification to 3 at the end of this period, which was reached 96 hours later. The term to describe these updraughts, 'vortical hot towers', was first coined by Hendricks, Montgomery, and Davis, 2004, and is still used today. Observational evidence of them is presented in studies as Houze Jr, Lee, and Bell, 2009, who provided detailed radar data of a specific 10 km wide vortical hot tower in the

formative stage of hurricane *Ophelia* (2005) from an aircraft mission. Montgomery states that vortical hot towers are accepted nowadays as the main drivers of tropical cyclone intensification, and are for a large part responsible for the axial asymmetry in tropical cyclones. Now the question arises whether the previous axisymmetric paradigms do still provide useful tools for understanding tropical cyclone intensification. In the next subsection, an axisymmetric view of the new rotating-convective updraught paradigm is presented and related to the previous paradigms.

### 2.2.5 An axisymmetric view of the new paradigm

Azimuthally averaged fields of the 2008 simulation from Sang et al. are provided by Smith, Montgomery, and Van Sang, 2009. They noted that spin up in a tropical cyclone occurs by two mechanisms and find that the axisymmetric view of Sang's 2008 model can be understood by a modified version of Ooyama's cooperative intensification paradigm. The first mechanism of spin up, which is already present in all previous paradigms, is the radial convergence of absolute angular momentum above the boundary layer by deep convection in the inner-core area of the cyclone. The second mechanism occurs inside the boundary layer. Although absolute angular momentum is not materially conserved within the boundary layer, the highest tangential velocities in the whole tropical cyclone are observed in the boundary layer. Air parcels high in the boundary layer experience little friction while they move quickly to smaller radii, $r$. Hence their tangential velocity increases fast due to approximate conservation of absolute angular momentum. This explains observed tangential wind speeds near the top of the boundary layer that are greater than the gradient wind, for example in Zhang et al., 2011. An interesting research whether azimuthally averaged fields of the 2008 simulation from Sang et al. can be captured by axisymmetric balance theory is performed by Bui et al., 2009. Here they forced a regularized version of the Sawyer-Eliassen equation with azimuthally-averaged fields from the non-symmetric model and compared the secondary (overturning) circulation and the primary circulation tendency. They found generally a good agreement in the results, except in the boundary layer where the balanced approximation does not hold.

## 2.3 Geostrophic adjustment

The theory of geostrophic adjustment describes the behaviour of a geostrophically balanced fluid under variations of the mass field due to heating. Geostrophic balance indicates the steady state in which the Coriolis force acting on the fluid is counteracted by the pressure gradient force. The response of the fluid to perturbations depends on the scale of the perturbations. The Rossby radius of deformation separates dynamically large from dynamically small perturbations. If the perturbation is dynamically large, there will be little potential energy release. However, the potential energy that is released will be easily converted into kinetic energy in the newly balanced state. If the perturbation is dynamically small, much of the potential energy is released. The released potential energy does not get easily converted into kinetic energy in the final state.

Middleton, 1987 researched geostrophic adjustment in a barotropic fluid following changes in the sea level. He found that the maximum kinetic energy in the balanced state following adjustment was obtained for sea level perturbations with a

typical size equal to the internal Rossby radius of deformation. This is in agreement with the above mentioned conceptual idea of gradient wind adjustment, as dynamically small perturbations do not convert much of their energy into the kinetic energy of the final balanced state, while dynamically large perturbations do not release much of their initial potential energy. Hence the largest kinetic energy in the new balanced state is obtained for horizontal scales in the order of the Rossby radius.

## 2.4 Gradient wind adjustment

Gradient wind balance is a force balance consisting of three horizontal components acting along the radial direction inside a tropical cyclone. The first component is the Coriolis force. This force represents the effect of the earth's rotation, acting in the outward direction. The second component is the pressure gradient force. Air parcels in a tropical cyclone experience a horizontal pressure gradient due to the pressure anomaly in the center. In most height levels in a tropical cyclone, the pressure anomaly is negative resulting in an inward force. The third component is the centrifugal force, which is the pseudo-force resulting from circular motion around the central vertical axis. Observational data show that the azimuthally averaged flow within a tropical cyclone core is close to gradient wind balance Willoughby, 1990.

As both the Coriolis force and the centrifugal force are pseudo-forces created by rotation of a non-inertial reference frame, it is possible to define a local Rossby radius of deformation that takes into account both rotational forces. Therefore, we may examine whether the intensification of a tropical cyclone can be understood as a process of gradient wind adjustment. In this process we expect the scale of the perturbation compared to the Rossby radius, to be important for the kinetic energy gain in the final balanced state. Results from a linear analysis of a modified version of Ooyama's 1969 model in Delden, 1989 (fig 13) show that intensification of a warm-cored tropical cyclone is strongly inhibited if the radius of the eye is much larger than the Rossby radius in the center of the cyclone.

# Chapter 3

# Model formulation

## 3.1 Dynamics

This section derives a theoretical model that will provide a basis for understanding the intensification of tropical cyclones by gradient wind adjustment, a process continuously occurring in cyclones due to latent heat release in moist convective cumulus clouds. We shall focus on a system consisting of two atmospheric layers plus

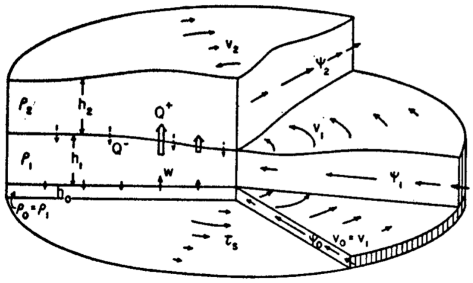

FIGURE 3.1: Ooyama's schematic diagram showing the basic design of the model.

boundary layer (of much larger horizontal than vertical extent), so we can assume the shallow water equations may be used. In essence, it is identical to the simplified 2-layer balanced model derived by Ooyama, 1969. His figure 1, depicting the schematics of the model, is shown in figure 3.1. Each layer has a constant density $\rho_j$, where $j$ represents the index of the boundary layer (0), bottom (1) or top (2) main layer. The densities are related to each other by

$$\rho_0 = \rho_1 = \rho \qquad\qquad \rho_2 = \epsilon\rho, \tag{3.1}$$

where $\epsilon$ is the hydrostatic stability parameter. The system is considered to be hydrostatically stable if $0 \leq \epsilon \leq 1$. To allow for vertical mass transfer from one main layer to the other, which represents diabatic heating/cooling in the real atmosphere, there is a net diabatic mass flux $Q$, consisting of an upward component and a downward component,

$$Q = Q^+ - Q^-. \tag{3.2}$$

A similar quantity for vertical transport between the boundary layer and bottom main layer is given by $w$, representing the vertical velocity at the top of the constant thickness boundary layer. Each layer has a thickness $h_j$ consisting of a (constant) standard value $\bar{h}_j$, which the thickness approaches at larger radii, and a spatially

and temporally varying perturbation $h'_j$.

$$h_j = \bar{h}_j + h'_j \tag{3.3}$$

The boundary layer has a constant thickness, so $h'_0 = 0$. Since the density is constant within the layers, the thickening rate of each layer is determined by the horizontal flow divergence and vertical mass flux at the interface of the main layers. The axisymmetric nature of this model implies that the only contribution to the divergence is due to the radial flow velocity. It is convenient to define a radial flux $\Psi_j$

$$\Psi_0 = -h_0 u_0 r \tag{3.4a}$$
$$\Psi_1 = -h_1 u_1 r \tag{3.4b}$$
$$\Psi_2 = -\epsilon h_2 u_2 r \tag{3.4c}$$

where $u_j$ represents the radial velocity in layer $j$, and $r$ the radial distance from the center. When $\Psi_j$ is multiplied by $\rho$, it represents the inward mass flux per unit arc across the entire thickness of layer $j$.

Now consider a fluid element at radius $r$ of height $h_j$, thickness $\delta r$ and width $r\delta\theta$. The total mass of this element is $m = \rho_j h_j r \delta r \delta\theta$. The contribution of the radial convergence to the time rate of change of mass of this fluid element can be written as $\dot{m}_{j,rad} = \rho\big[\Psi_j(r+\delta r) - \Psi_j(r)\big]\delta\theta \approx \rho\frac{\partial \Psi_j}{\partial r}\delta r\delta\theta$. The vertical mass transfer is $\rho w r \delta r \delta\theta$ and $\rho Q r \delta r \delta\theta$ for the boundary-bottom layer interface and bottom-top layer interface, respectively. We can now formulate the continuity equation in each layer by differentiating $m$ with respect to time and by adding the fluxes, the result is:

$$\frac{\partial h_0}{\partial t} = \frac{\partial \Psi_0}{r\partial r} - w = 0 \tag{3.5a}$$
$$\frac{\partial h_1}{\partial t} = \frac{\partial \Psi_1}{r\partial r} - Q + w \tag{3.5b}$$
$$\epsilon\frac{\partial h_2}{\partial t} = \frac{\partial \Psi_2}{r\partial r} + Q \tag{3.5c}$$

Throughout the main part of the cyclone above the boundary layer, we assume hydrostatic balance. Within the boundary layer, the pressure is constant with respect to height. The governing equations for the pressure $p_j$ in the different layers are therefore given by

$$p_0 = p_1 \tag{3.6a}$$
$$p_1 = \rho g(h_1 + \epsilon h_2 - z) + p_a \tag{3.6b}$$
$$p_2 = \epsilon\rho g(h_1 + h_2 - z) + p_a \tag{3.6c}$$

where $p_a$ denotes the ambient pressure at the top of the upper layer, and $g$ is the gravitational constant. Now let $\bar{p}_j$ be the standard pressure corresponding to the undisturbed thickness $\bar{h}_j$ of each layer at large radius, with this we can define the deviation of the geopotential $\phi_j$:

$$\phi_j = \frac{p_j - \bar{p}_j}{\rho_j} \tag{3.7}$$

Substituting the pressure equations for all layers we get

$$\phi_0 = \phi_1 \tag{3.8a}$$
$$\phi_1 = g[(h_1 - \bar{h}_1) + \epsilon(h_2 - \bar{h}_2)] \tag{3.8b}$$
$$\phi_2 = g[(h_1 - \bar{h}_1) + (h_2 - \bar{h}_2)] \tag{3.8c}$$

In this model we assume that the primary or tangential circulation in all layers is balanced by the three acting forces: the Coriolis force, the centrifugal force and the pressure gradient force. This balance is given by the gradient wind equation

$$fv_j + \frac{v_j^2}{r} - \frac{\partial \phi_j}{\partial r} = 0, \tag{3.9}$$

where $f$ represents the Coriolis parameter, and $v_j$ the tangential velocity. This balance dictates a constraint on the swirling flow of the tropical cyclone. The absolute angular momentum, $M_j$, per unit mass consists of the relative momentum of the fluid and the added momentum by the rotation of the earth's surface. It is defined as

$$M_j = v_j r + \frac{1}{2} f r^2 \tag{3.10}$$

From equations (3.8, 3.9, 3.10), we deduce that

$$v_0 = v_1 \qquad\qquad M_0 = M_1 \tag{3.11}$$

At the ocean-air interface, the angular momentum flux, $Z_s$, (apart from a factor $\rho$) produced by the tangential shearing stress, $\tau_s$, is equal to

$$Z_s = -\frac{\tau_s r}{\rho}, \tag{3.12}$$

where $\tau_s$ itself is given by

$$\tau_s = \rho C_D |v_1| v_1 \quad \text{with} \quad C_D = (0.5 + 0.06 v_1) \times 10^{-3}, \tag{3.13}$$

with $v_1$ measured in m/s. The drag coefficient $C_D$ is adjusted for use with the gradient wind, instead of the usual (lower) anemometer level wind. The net vertical transport of angular momentum from the boundary layer to the bottom main layer, $Z_{01}$, and from the bottom main layer to the top main layer, $Z_{12}$, are given by

$$Z_{01} = M_1 w \tag{3.14a}$$
$$Z_{12} = M_1 Q^+ - M_2 Q^- + \mu(M_1 - M_2), \tag{3.14b}$$

where $\mu$ denotes the linear coefficient of tangential shearing stress at the bottom-top main layer interface. The radial flux of angular momentum per unit arc by lateral eddy transport $\Lambda_j$ is calculated with

$$\Lambda_1 = \lambda_1 h_1 r^3 \frac{\partial}{\partial r}\left(\frac{v_1}{r}\right) \tag{3.15a}$$
$$\Lambda_2 = \epsilon \lambda_2 h_2 r^3 \frac{\partial}{\partial r}\left(\frac{v_2}{r}\right), \tag{3.15b}$$

where $\lambda_j$ denotes the (kinematic) coefficient of eddy viscosity. The angular momentum budget of the layers is then determined by

$$\frac{\partial h_0 M_0}{\partial t} = \frac{\partial}{r\partial r}(\Psi_0 M_1) - Z_{01} + Z_s \approx 0 \tag{3.16a}$$
$$\frac{\partial h_1 M_1}{\partial t} = \frac{\partial}{r\partial r}(\Psi_1 M_1 + \Lambda_1) - Z_{12} + Z_{01} \tag{3.16b}$$
$$\epsilon\frac{\partial h_2 M_2}{\partial t} = \frac{\partial}{r\partial r}(\Psi_2 M_2 + \Lambda_2) + Z_{12} \tag{3.16c}$$

The continuity equation (3.5) and definition of the geopotential (3.8) can be combined into

$$\frac{\partial \phi_1}{\partial t} = g\frac{\partial}{r\partial r}(\Psi_1 + \Psi_2) + G_1 \tag{3.17a}$$
$$\frac{\partial \phi_2}{\partial t} = g\frac{\partial}{r\partial r}(\Psi_1 + \epsilon^{-1}\Psi_2) + G_2, \tag{3.17b}$$

where

$$G_1 = gw \tag{3.18a}$$
$$G_2 = g(w + \epsilon^{-1}(1-\epsilon)Q). \tag{3.18b}$$

Now we can also combine the continuity equation (3.5) with the angular momentum budget (3.16) after which we end up with

$$\frac{\partial v_1 r}{\partial t} = h_1^{-1}(f + \zeta_1)\Psi_1 + F_1 \tag{3.19a}$$
$$\frac{\partial v_2 r}{\partial t} = (\epsilon h_2)^{-1}(f + \zeta_2)\Psi_2 + F_2, \tag{3.19b}$$

where

$$F_1 = h_1^{-1}\left[(Q^- + \mu)(v_2 - v_1)r + \frac{\partial \Lambda_1}{r\partial r}\right] \tag{3.20a}$$
$$F_2 = (\epsilon h_2)^{-1}\left[(Q^+ + \mu)(v_1 - v_2)r + \frac{\partial \Lambda_2}{r\partial r}\right] \tag{3.20b}$$

and for $j = 1, 2$, the relative vorticity, $\zeta_j$, is given by

$$\zeta_j = \frac{\partial(v_j r)}{r\partial r}. \tag{3.21}$$

Equation 3.16a may be written simply as

$$\Psi_0 = (f + \zeta_1)^{-1} C_D |v_1| v_1 r \tag{3.22}$$

The radial flux $\Psi_j$ in the main layers must be determined such that it supports the vortex in maintaining a state of gradient wind balance. In order to do this, the gradient wind balance equation (3.9) is differentiated with respect to time and time derivatives are consequently eliminated using the prognostic equations of the geopotential (3.17) and tangential velocity (3.19) to obtain the following condition

for the radial flux

$$r\frac{\partial}{\partial r}\left(\frac{\partial}{r\partial r}(\Psi_1+\Psi_2)\right)-S_1\Psi_1=B_1 \tag{3.23a}$$

$$r\frac{\partial}{\partial r}\left(\frac{\partial}{r\partial r}(\Psi_1+\epsilon^{-1}\Psi_2)\right)-\epsilon^{-1}S_2\Psi_2=B_2 \tag{3.23b}$$

where,

$$S_j=\left(f+\frac{2v_j}{r}\right)\left(\frac{f+\zeta_j}{gh_j}\right) \tag{3.24a}$$

$$B_j=\frac{1}{g}\left[\left(f+\frac{2v_j}{r}\right)F_j-r\frac{\partial}{\partial r}G_j\right] \tag{3.24b}$$

The boundary conditions for solving equation 3.23 are given by

$$\begin{cases}\Psi_1=\Psi_2=0, & \text{at } r=0\\ \Psi_0+\Psi_1+\Psi_2=0, & \text{at } r=r^*\\ \dfrac{\partial\Psi_2}{\partial r}=-\dfrac{\Psi_2}{R}, & \text{at } r=r^*\end{cases} \tag{3.25}$$

where $r^*$ denotes the maximum radius at which the model is evaluated, and $R$ the Rossby radius of (internal) deformation.

## 3.2 Convection

So far we have constructed a dynamical foundation of the numerical model which is able to evolve in time given a certain diabatic heating. In this section, the closure that is incorporated to link the diabatic heating to the resolved-scale dynamics will be elucidated. Ooyama, 1969 noted that the majority of the diabatic heating occurs in tall convective towers (vortical hot towers), and that observational evidence supported that the amount of activity of the convection towers is well-connected to the convergence of the boundary layer inflow. Hence, the upward vertical mass flux between the two main layers $Q$ in this model is proportional to the vertical velocity at the top of the boundary layer $w$. We will assume that $Q^+$ is given in the form:

$$Q^+=\begin{cases}\eta w, & \text{if } w>0\\ 0, & \text{if } w\leq 0.\end{cases} \tag{3.26}$$

$Q^-=0$ in this research, as there has not been made any attempt to include the effect of diabatic cooling. Let us define $H$ as the sum of the enthalpy including latent heat and geopotential energy

$$H=c_pT+Lq+gz. \tag{3.27}$$

Here, $c_p$ denotes the specific heat of air at constant pressure, $T$ the temperature, $L$ the latent heat, $q$ the mixing ratio and $z$ the vertical height. In differential form it may be expressed as:

$$dH=\left(\frac{c_pT}{\theta_e}\right)d\theta_e, \tag{3.28}$$

where $\theta_e = \theta \exp\left(\frac{Lq_s}{c_p T}\right)$ represents the equivalent potential temperature, with $q_s$ the saturation mixing ratio. During pseudo-adiabatic ascent, we assume $H$ to be a conserved property of an air parcel. If we let $H_0, H_1$ and $H_2$ be representative values of $H$ in the boundary layer, bottom and top main layer, we can construct an energy conservation equation:

$$\eta H_2 = (1-\eta)H_1 + H_0, \tag{3.29}$$

or, after rewriting,

$$\eta = 1 + \frac{H_0 - H_2}{H_2 - H_1}. \tag{3.30}$$

Now assuming the variations in $\theta_e$ and $H$ are small, we may assume the factor in parentheses in equation 3.28 to be constant, so that $H$ can easily be expressed in terms of $\theta_e$. Representative values of $\theta_e$ for the boundary layer and bottom main layer are $(\theta_e)_0$ and $(\theta_e)_1$. In the upper layer, it will represent the saturated cloud air, so we use the saturated equivalent potential temperature $(\theta_e^*)_2$. For simplicity, we will make use of the following $\chi$-notations:

$$\chi_0 = (\theta_e)_0 - \Theta \tag{3.31a}$$
$$\chi_1 = (\theta_e)_1 - \Theta \tag{3.31b}$$
$$\chi_2 = (\theta_e^*)_2 - \Theta, \tag{3.31c}$$

where $\Theta$ represents a reference equivalent potential temperature. Equation 3.30 may be rewritten to

$$\eta = 1 + \frac{\chi_0 - \chi_2}{\chi_2 - \chi_1} \tag{3.32}$$

Now we will define a mean potential temperature $\theta_m$, which satisfies thermal wind balance:

$$(\Pi_1 - \Pi_2)\frac{\partial \theta_m}{\partial r} = \left(f + \frac{v_2}{r}\right)v_2 - \left(f + \frac{v_1}{r}\right)v_1 \tag{3.33}$$

with

$$\Pi_j = c_p \left(\frac{p_j}{p_{ref}}\right)^{2/7} \tag{3.34}$$

using a reference pressure $p_{ref}$ of 1000 hPa. Inserting equation 3.9, we then get

$$\theta_m - \bar{\theta}_m = \frac{\phi_2 - \phi_1}{\Pi_1 - \Pi_2}, \tag{3.35}$$

where $\bar{\theta}_m$ represents the value that $\theta_m$ approaches at large radii, where $\phi_j$ approaches zero. We will now assume that the temperature profile in the main layers is very close to that of a moist adiabat, equal to $(\theta_e^*)_2$. Therefore $\theta_m$, the average of $\theta_1$ and $\theta_2$, is defined uniquely for specific values of $(\theta_e^*)_2$. Ooyama uses the following interpolation formula for typical observed $\theta_e$ values:

$$\chi_2 - \bar{\chi}_2 = 2(\theta_m - \bar{\theta}_m) \tag{3.36}$$

or, rewritten using equation 3.35,

$$\chi_2 - \bar{\chi}_2 = \alpha c_p^{-1}(\phi_2 - \phi_1), \tag{3.37}$$

where $\alpha$ is a parameter roughly equal to 10 for typical circumstances. The energy budget in the boundary layer may be written as

$$\frac{\partial}{\partial t}(h_0 H_0) - \frac{\partial}{r\partial r}(\Psi_0 H_0) + (w^+ H_0 - w^- H_1) = C_E |v_1|(H_s - H_0) \tag{3.38}$$

where $H_s$ denotes the values of $H$ for saturated air at sea level pressure and temperature. $w^+$ and $w^-$ are defined by

$$w^+ = 0.5(|w| + w) \tag{3.39a}$$
$$w^- = 0.5(|w| - w) \tag{3.39b}$$
$$, \tag{3.39c}$$

and $C_E$ represents the coefficient for the air-sea energy exchange. In the current research, $C_E$ is equal to the drag coefficient $C_D$. The energy budget may be rewritten in terms of $\chi$'s as

$$\frac{\partial \chi_0}{\partial t} + u_0 \frac{\partial \chi_0}{\partial r} + \frac{w^-}{h_0}(\chi_0 - \chi_1) = C_E \frac{|v_1|}{h_0}(\chi_s - \chi_0) \tag{3.40}$$

where

$$\chi_s = (\theta_e^*)_s - \Theta \tag{3.41}$$

with $(\theta_e^*)_s$ the equivalent potential temperature of the saturated air near the sea surface. The effect of surface pressure on the saturated equivalent potential temperature is captured by

$$\chi_s - \bar{\chi}_s = -\beta c_p^{-1} \phi_1, \tag{3.42}$$

where $\beta$ takes an approximate value of 2.

## 3.3 Discretization

The equations derived in the previous sections are approximated by discrete functions. Variables in the numerical model are defined either on a regular radial grid, $r^k = k\Delta r$ for $k = 0, 1, ..., K-1$ of length $K$, or on a staggered radial grid $r^{k'} = k'\Delta r = (k + \frac{1}{2})\Delta r$ for $k = 0, 1, ..., K-2$ of length $K-1$. Variables that are non-zero at the origin are typically defined on the staggered grid.

Some important variables that are defined on the regular grid are $v$ and$\Psi$. Staggered grid variables include $\phi, h, Q, w, \eta$ and all the $\chi$'s. In the `Python` model, all variable names of discrete variables on the staggered grid are distinguishable by an underscore (_) in front of them, e.g. $\_\phi$. The complete model is shown in appendix A.

In some calculations, variables on the staggered grid need to be defined on the regular grid, or the other way round. This conversion is done using a weighted average. A weighted averaged of function $\_f$ defined on the staggered grid is shown below:

$$[\_f]_k = (r^{k'} \_f^{k'} + r^{k'-1} \_f^{k'-1})/2r^k.$$

The $[]_k$-notation is used to denote the weighted average. We can similarly tranform a function $f$ on the regular grid to the staggered grid:

$$[f]_{k'} = (r^k f^k + r^{k+1} f^{k+1})/2r^{k'}.$$

All discrete variables, are defined on a dynamical time grid of length $N$. The time step, $\Delta t$, is adjusted depending on the radial velocity, $u_0$, in the boundary layer. During the integration, the quantity $\delta = \max(|u_0|)\frac{\Delta t}{\Delta r}$ is evaluated. If $\delta > 0.5$, $\Delta t$ is decreased such that $\delta = 0.4$, else, if $\delta < 0.4$, $\Delta t$ is increased such that $\delta$ becomes 0.5. In this way, $\Delta t$ is chosen as large as possible while avoiding numerical instability.

An example of the discretized equivalent of an equation with a first order radial derivative (eq. 3.9) is

$$(f + \frac{v_j^k}{r^k})v_j^k = \frac{\phi_j^{k'} - \phi_j^{k'-1}}{\Delta r}. \tag{3.43}$$

The radial derivative of $\_\phi$ at $r^k$ is approximated by a central difference method, using values at $r^{k'}$ and $r^{k'-1}$. Many radial derivatives of staggered grid variables are approximated at the regular grid with a central difference method in the same way. The same central method can be applied for regular grid variables whose derivative must be specified on the staggered grid.

The prognostic equation for $v_j$ (3.19) is used to make a prediction of $v_j$ before $\phi_j$ at the new time step is actually calculated. The discretized version is shown here:

$$\frac{v_1^{k,n+1} - v_1^{k,n}}{\Delta t} r^k = \frac{f + < \zeta_1^n >_k}{[h_1^n]_k} \Psi_1^{k,n} + F_1^{k,n} \tag{3.44a}$$

$$\frac{v_2^{k,n+1} - v_2^{k,n}}{\Delta t} r^k = \frac{f + < \zeta_2^n >_k}{\epsilon[h_2^n]_k} \Psi_2^{k,n} + F_2^{k,n} \tag{3.44b}$$

where

$$F_1^{k,n} = [h_1^n]_k^{-1}[(Q^-)^n]_k(v_2^{k,n} - v_1^{k,n})r$$
$$F_2^{k,n} = (\epsilon[h_2^n]_k)^{-1}[(Q^+)^n]_k(v_1^{k,n} - v_2^{k,n})r$$

and

$$< \zeta_j >_k^n = \begin{cases} \frac{v_j^{k+1,n} r^{k+1} - v_j^{k,n} r^k}{r^{k'} \Delta r} & \text{if } \Psi_j^{k,n} > 0 \\ \frac{v_j^{k,n} r^k - v_j^{k-1,n} r^{k-1}}{r^{k'-1} \Delta r} & \text{if } \Psi_j^{k,n} < 0. \end{cases}$$

Here the $[]_k$ notation is used to indicate the weighted average taken to evaluate variables at regular grid points. The calculation of $\zeta$ is adjusted to use an upstream scheme, that utilizes the value of $\Psi_j$ to determine what direction upstream is for every grid point. Note that the calculation of $\zeta$ can also involve an estimated value of $v$ for a split-level uncentered upstream difference method. This is formulated in the parts of equation A7 in Ooyama's paper.

The radial mass flux equation for the main layers involves a bit more algebra in deriving its discrete version. This derivation is therefore performed below in more detail. We start out with equation 3.23:

$$r\frac{\partial}{\partial r}\left[\frac{\partial}{r\partial r}(\Psi_1 + \Psi_2)\right] - S_1\Psi_1 = B_1 \tag{3.45a}$$

$$r\frac{\partial}{\partial r}\left[\frac{\partial}{r\partial r}(\Psi_1 + \epsilon^{-1}\Psi_2)\right] - \epsilon^{-1}S_2\Psi_2 = B_2 \tag{3.45b}$$

We may assume for now that $S_j$ and $B_j$ are known at regular radial grid points. For the sake of concise description we will use the abbreviation $\delta_r$ instead of $\frac{\partial}{\partial r}$. With

this, the radial flux equation becomes:

$$\begin{aligned} r\delta_r r^{-1}\delta_r(\Psi_1+\Psi_2) - S_1\Psi_1 &= B_1 \\ r\delta_r r^{-1}\delta_r(\Psi_1+\epsilon^{-1}\Psi_2) - \epsilon^{-1}S_2\Psi_2 &= B_2 \end{aligned}$$

$$\begin{aligned} \left(\delta_r^2 - \frac{1}{r}\delta_r\right)(\Psi_1+\Psi_2) - S_1\Psi_1 &= B_1 \\ \left(\delta_r^2 - \frac{1}{r}\delta_r\right)(\Psi_1+\epsilon^{-1}\Psi_2) - \epsilon^{-1}S_2\Psi_2 &= B_2 \end{aligned}$$

$$\begin{aligned} \left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_1 - S_1\Psi_1 &= B_1 - \left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_2 \\ \left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_2 - S_2\Psi_2 &= \epsilon B_2 - \epsilon\left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_1 \end{aligned}$$

$$\begin{aligned} (1-\epsilon)\left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_1 - S_1\Psi_1 &= B_1 - \epsilon B_2 - S_2\Psi_2 \\ (1-\epsilon)\left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_2 - S_2\Psi_2 &= \epsilon\left(B_2 - B_1 + S_1\Psi_1\right) \end{aligned}$$

The radial changes in $\Psi_j$ are important in all equations involving the conservation of mass, since these changes are an integral part of the mass budget equation for each layer. Since this is the case, we might solve the above equation using an estimation of $\Psi_j$ for the $\Psi_j$ terms on the right hand side of the equations, and calculate the radial distribution accordingly by solving for $\Psi_j$ on the left hand side. For the sake of concise description, let's define the right hand side of both equations to be a forcing term $A_j$ which is known at all regular grid points such that

$$(1-\epsilon)\left(\delta_r^2 - \frac{1}{r}\delta_r\right)\Psi_j - S_j\Psi_j = A_j$$

Now we can convert to a discrete version of these equations using central difference methods on the regular radial grid. The result is shown below:

$$\frac{\Psi_j^{k+1} - 2\Psi_j^k + \Psi_j^{k-1}}{\Delta r^2} - \frac{\Psi_j^{k+1} - \Psi_j^{k-1}}{2r^k\Delta r} - (1-\epsilon)^{-1}S_j^k\Psi_j^k = (1-\epsilon)^{-1}A_j^k \tag{3.46}$$

Since $r^k = k\Delta r$, we can rewrite this into

$$\left(\frac{k-\frac{1}{2}}{k}\right)\Psi_j^{k+1} - \left(2 + (1-\epsilon)^{-1}\Delta r^2 S_j^k\right)\Psi_j^k + \left(\frac{k+\frac{1}{2}}{k}\right)\Psi_j^{k-1} = (1-\epsilon)^{-1}\Delta r^2 A_j^k$$

The above is a linear system of equations $M_j\Psi_j = (1-\epsilon)^{-1}\Delta r^2 A_j$. One way to find the solution is by successive over-relaxation. Such a method has been succesfully implemented, however it was later replaced by the `numpy.linalg.solve()` function due to computational efficiency reasons.

The numerical integration of the model is explained in Ooyama, 1969. Interesting to note is the fact that before the radial mass budget, shown above, is solved using

accurate estimates of $v$ and $\zeta$. Without these accurate values, the numerical integration quickly becomes unstable as many processes depend directly on the convergence of $\Psi$. The time step for integration is determined dynamically as mentioned earlier this section. The evolution of the $\chi$'s is calculated in a separate scheme. As there is an advection term with the boundary layer velocity in this calculation, the time step is also adjusted dynamically in the same way. Because the radial velocity in the boundary layer is quite high, the time step of integrating the equation for the $\chi$'s is usually smaller than the time step of the integration of $v$ and *phi*.

## 3.4 Comparison

In order to evaluate the credibility of the numerical model, a test run was made to reproduce the results in Ooyama, 1969. The radial distributions of different variables are shown in figure 3.2 and 3.3. The run corresponds to case A in Ooyama, 1969. The original results are shown in figures 3.4 and 3.5. Both figures have the same structure. The different columns represent variables after certain periods of time after initialization, indicated in the right corner of the plots in the top row of the figure. The top row shows the tangential wind in both main layers and the inward radial velocity in the boundary layer. After 47, 81, 108 and 134 hours, the tangential velocity sharply increases attaining a maximum at around 134 hours. After 162 and 194 hour we see that the maximum wind attained is decreasing in both layers, but the profile broadens significantly. One interesting feature is that the tangential wind in the top main layer is always smaller than the tangential wind in the bottom main layer, and above a certain radius is anticyclonic. The boundary layer inflow is positive everywhere, with a maximum at a radius usually some kilometers greater than the radius of maximum wind.

The second row shows the inward radial mass flux $\Psi$ for the boundary layer and the sum of the boundary layer and bottom main layer. The radial mass flux for the top main layer is not shown since it deviates very little from the sum of the boundary layer and lower main layer in absolute value, in the order of $100\,\mathrm{m}^3/\mathrm{s}$, which is in agreement with the findings of Ooyama, 1969. The radial mass flux increases during the course of the integration, as the frictional inflow increases.

The third row shows the radial distribution of $w$ and $\eta$. $w$ is close to zero nearly everywhere but inside the first 100 km from the center of the cyclone. Especially during the first three time steps, its distribution is very localized near the eyewall, not reaching much further than about 60 km from the center. In these time steps we observe a rapid growth in peak intensity up to about $4\,\mathrm{m/s}$. After $t = 134\,\mathrm{hr}$ the peak intensity diminishes but the total radial interval covered by the vertical motion increases. $\eta$ obtains relatively high values in the center of the cyclone, and low values where the upward velocity has a maximum in the first three time steps. This profile is mainly due to the radial distribution of $\chi_0$, which is shown in the row below. While $\eta$ is initialized with a value of 2 at all radii, during the course of the integration there is a downward trend toward a value of 1, starting from the center of the cyclone, while continuing on larger radii as the cyclone intensifies.

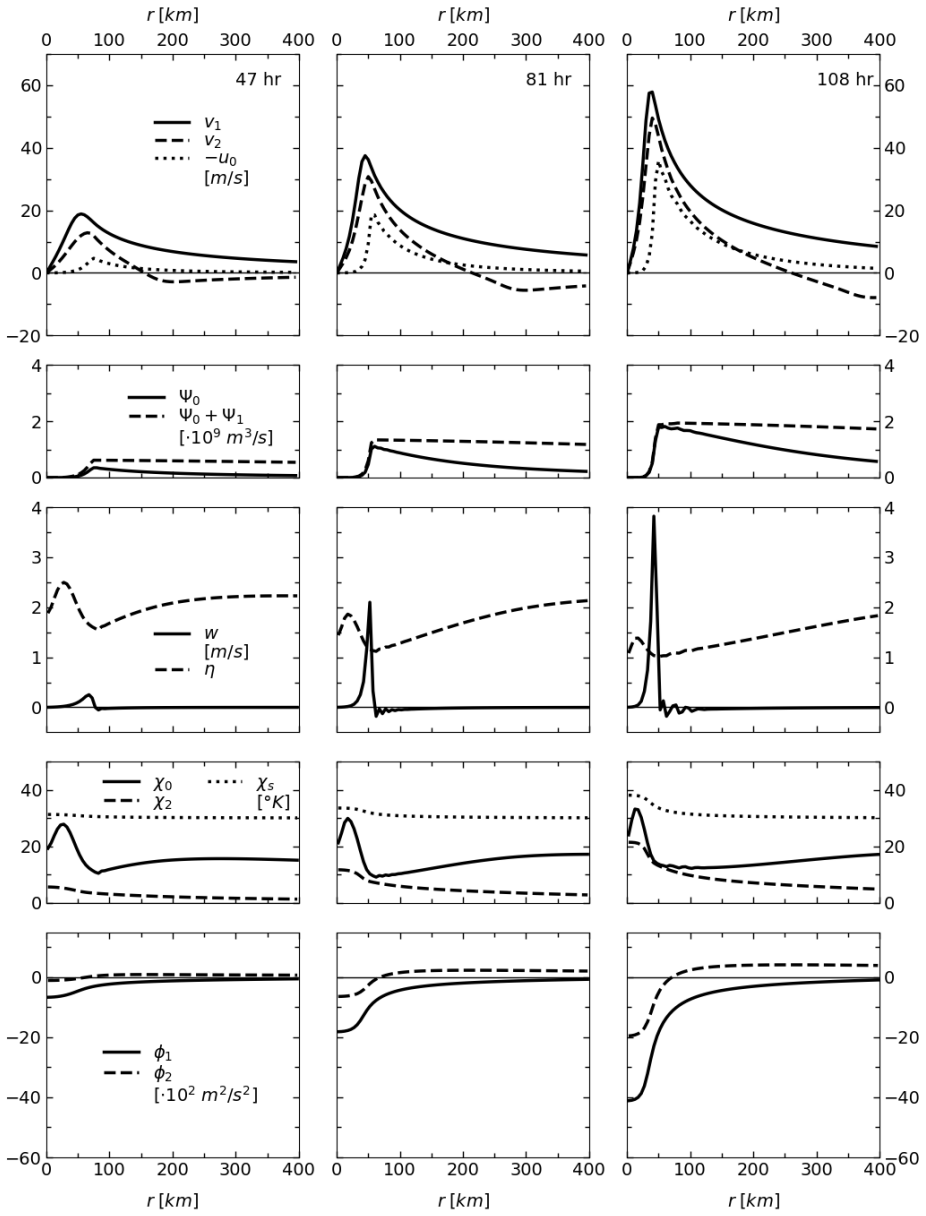


FIGURE 3.2: The numerical model output of variables as function of $r$ at 47, 81 and 108 hours after initialization with a vortex with $v_{max} = 10\,\mathrm{ms}^{-1}$

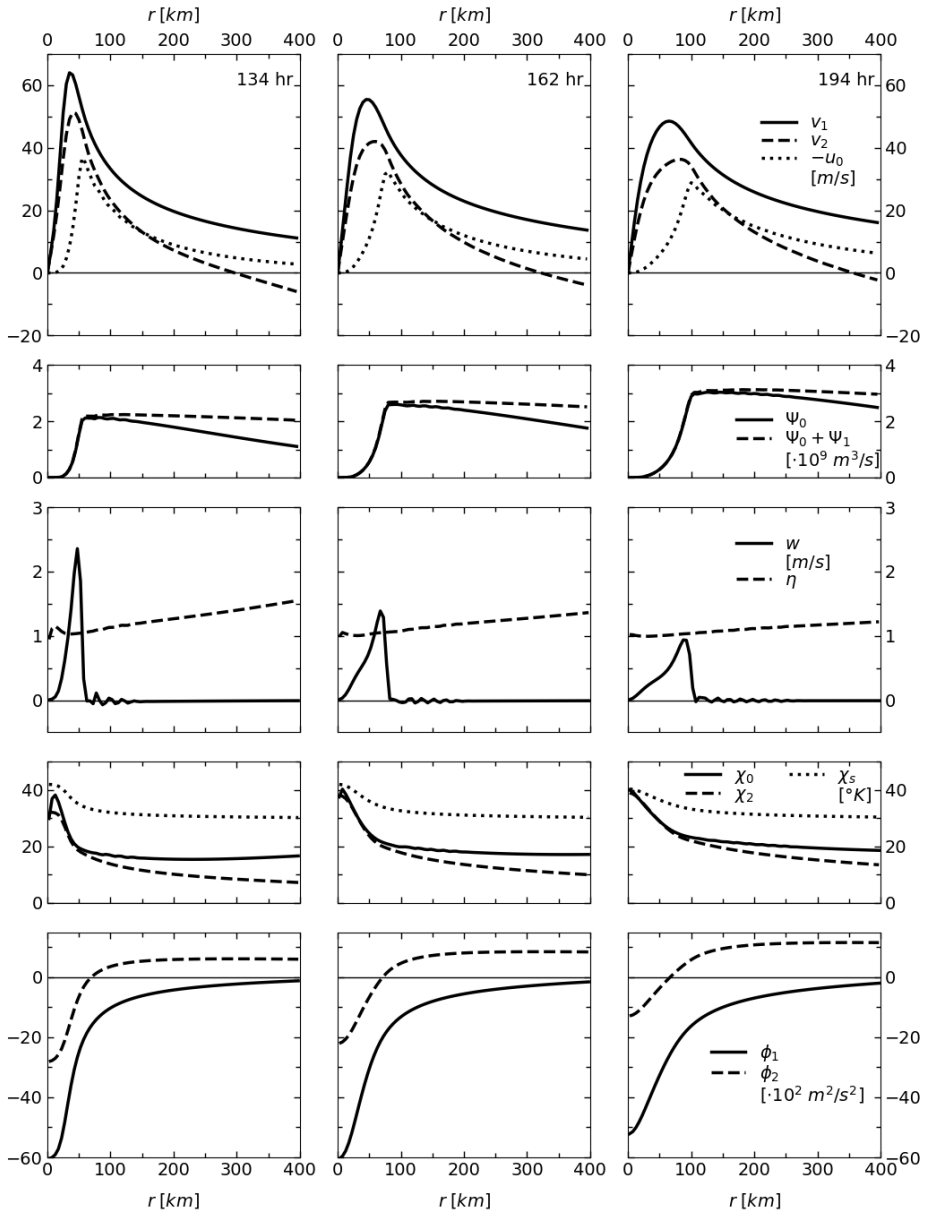


FIGURE 3.3: The numerical model output of variables as function of $r$ at 134, 162 and 194 hours after initialization with a vortex with $v_{max} = 10\,\mathrm{ms}^{-1}$

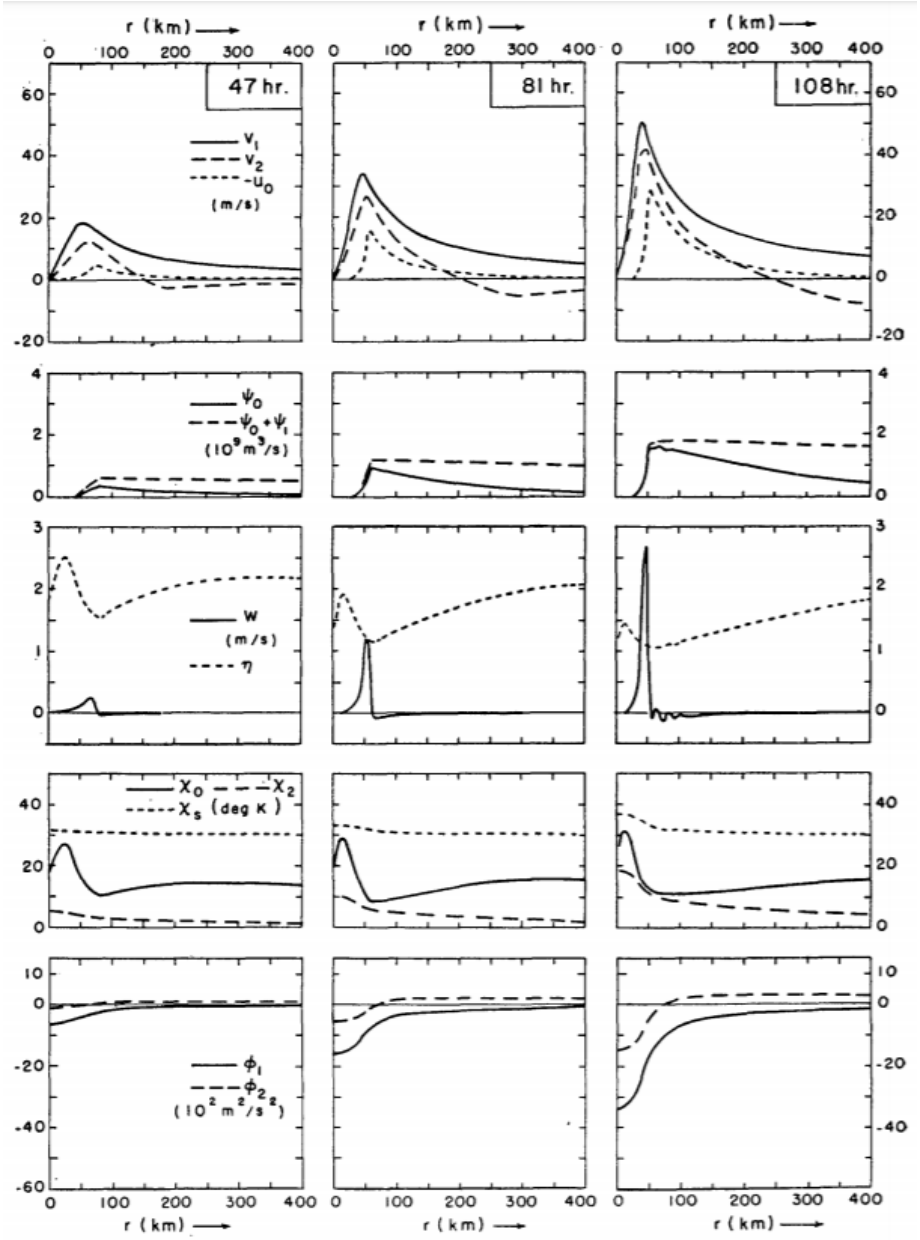


FIGURE 3.4: Ooyama's original numerical model output of variables as function of $r$ at 47, 81 and 108 hours after initialization with a vortex with $v_{max} = 10\,\mathrm{ms}^{-1}$

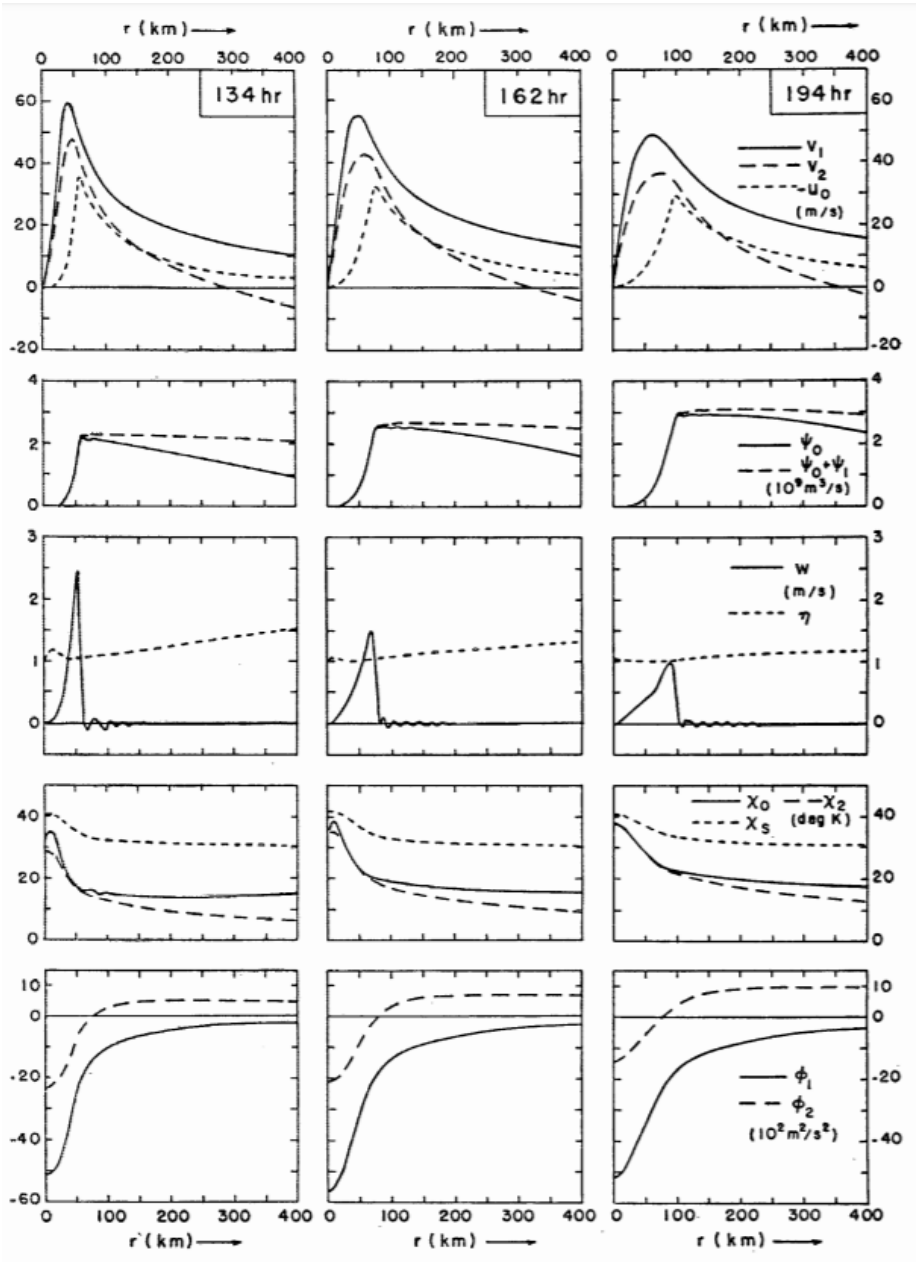


FIGURE 3.5: Ooyama's original numerical model output of variables as function of $r$ at 134, 162 and 194 hours after initialization with a vortex with $v_{max} = 10\,\mathrm{ms}^{-1}$

The fourth row shows the radial distribution of the $\chi$'s. Since with the current parameters $\chi_s \geq \chi_0$, we conclude that the sea surface adds sensible and latent heat to the boundary layer flow. $\chi_0$ in general seems to decrease during the deepening stage, except for the region confined within the outer radius of the eyewall, where it increases. The general decrease at larger radii may be expected since the rapidly decreasing central surface pressure is responsible for more divergence in the boundary layer flow, which is accounted for by subsidence of colder and drier air. Indeed the plot of $w$ shows slightly negative values in this range. The numerical noise in $w$ is also visible in the plot of $\chi_0$, which is to be expected as this subsidence influences the equivalent potential temperature anomaly. The increasing value of $\chi_0$ within the outer radius of the eyewall occurs since the saturated warm air is supplied by the

radial inflow, yet there is little outflow in the vertical direction, hence the amount of sensible heat in this region is increasing with time. $\chi_2$ also increases during this period, especially in the central area, due to intrusion of warm air. After $t = 134$ hr, the cyclone enters the filling stage. During the filling stage $\chi_2$ approaches $\chi_0$, mainly in the region of high $w$. This results in $\eta$ being locally close to unity, such that there is little entrainment of air from the bottom main layer in the convective updraught. Since spin up of the vortex in the numerical model is mainly achieved by radial inflow of air in the bottom main layer, the intensification in the bottom main layer in terms of the tangential wind is inhibited where there is no entrainment ($\eta = 1$).

The fifth row shows the geopotential anomaly in both main layers. The geopotential anomaly of the bottom main layer increases with $r$ at all radii, indicating cyclonic flow as you expect in the lower troposphere. In the top main layer, the central region has a positive $\phi$-gradient, whereas the outer region has a negative one, which is to be expected looking at the tangential velocity profile. During the first four time steps we see a clear deepening of the geopotential near $r = 0$, while for the latter two time steps the central area of low geopotential expands to larger radii and becomes less deep.

Figure 3.6 shows the temporal evolution of the central pressure anomaly, the central surface pressure tendency, the tangential wind maxima in the main layers, the tangential wind minimum in the top layer, the radius of maximum wind, $r_0$, the Rossby radius, evaluated at the radius of maximum heating, the Rossby radius evaluated at 50 km from the cyclone center, the energy conversion ratio and the total kinetic energy.

During the first 75 hours, we observe a moderate increase of the maximum tangential wind and a moderate decrease of the central surface pressure. This is followed by a sharp increase in tangential velocity and rapidly decreasing central surface pressure for the next 50 hours, approximately. Finally, the central surface pressure starts to increase again while the maximum tangential wind decreases. Although the maximum tangential wind decreases, this does not mean that the cyclone is weakening. In contrary, the total kinetic energy is growing even till the last time step of integration. Therefore we may note that the cyclone looses some kinetic energy near the radius of maximum wind, but strengthens at other radii. We see that the central surface pressure lags behind the maximum tangential wind by approximately one day. The radius of maximum wind is found to decrease during the period of fast decreasing central surface pressure, indicating a spin up process. Once the central surface pressure increases with increasing time the radius of maximum wind also increases, indicating the expanding behaviour of the vortex core during the filling stage.

During the filling period $r_0$ increases from roughly 50 to 100 km. The shift of $r_0$ is likely related to the increased central surface pressure tendency, which after roughly 135 hour even becomes positive. During the first 80 hours of integration, the energy conversion ratio increases linearly with time from just below 0.02 to 0.06. After this period the energy conversion ratio decreases again. The point of highest energy conversion ratio corresponds to the fastest growth in the maximum tangential wind speed and fastest decline in the central surface pressure, as is visible in the figure. Interestingly, $KE$ increases at a moderate pace during the first 70 hours, but at a much higher rate after this period. The increase after this period is also linear in behaviour. Since the energy conversion ratio is decreasing with time during this period, this must mean that the total amount of diabatic heating increases more than the amount necessary to overcome the loss of energy conversion efficiency.

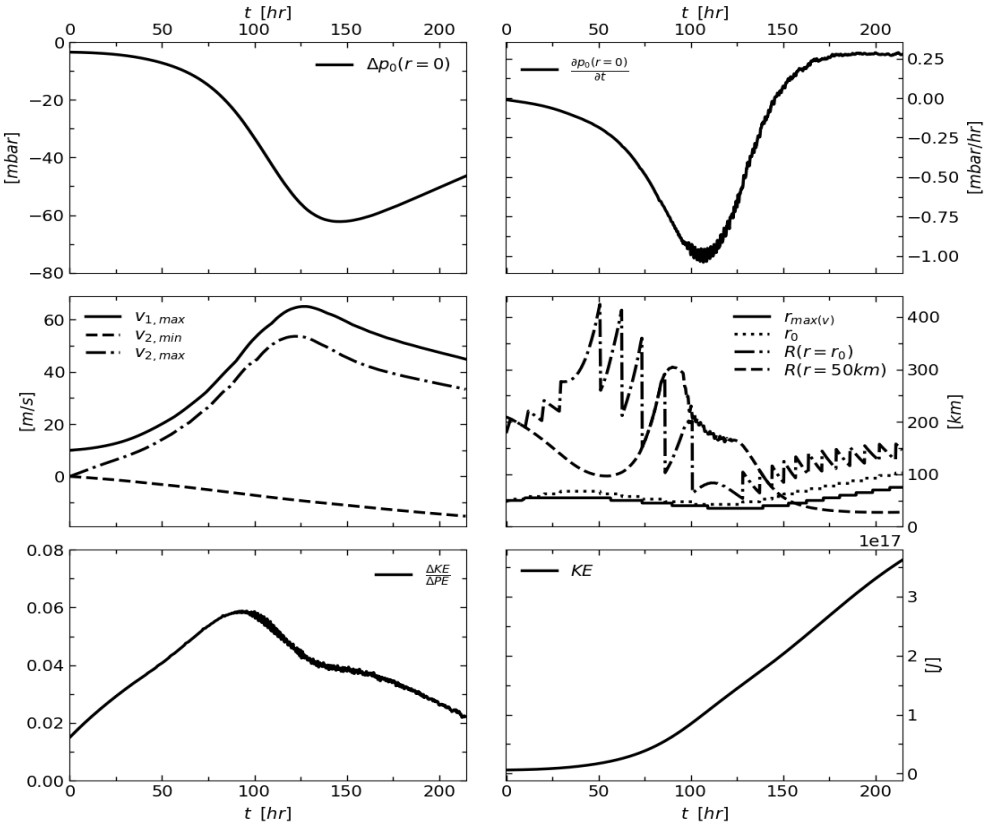


FIGURE 3.6: Model output of 1100 time steps. The shown variables as a function of time are the central surface pressure anomaly, $\Delta p_0$ (upper left), the central surface pressure tendency, $\frac{\partial p_0}{\partial t}$, (upper right), the maximum tangential velocity, $v$, of the bottom main layer and the minimum and maximum tangential velocities of the top main layer (center left), the radius of maximum tangential wind, $r_{max(v)}$, the radius of maximum heating, $r_0$, the Rossby radius of deformation, $R$, evaluated at $r = r_0$ and at $r = 50$ km (center right), the energy conversion ratio, $\frac{\Delta KE}{\Delta PE}$, (bottom left) and the kinetic energy $KE$, (bottom right)

Looking at the present results and those of Ooyama, 1969, we see a lot of similarity. The tropical cyclone intensifies mainly near the center during its deepening stage, matures and finally intensifies at larger radii with a simultaneous decrease of the maximum tangential wind near the center during the filling stage. Quantitatively, there are some minor differences. For example the maximum of $w$ at 108 hour is more than 1 m/s more than in the original paper. The minimum of $\phi_1$ differs about 10 mbar. In this way, practically all variables differ a little bit from the original values found by Ooyama, 1969. The maximum tangential wind and minimum central surface pressure are found slightly earlier in time than in the original model outcomes as well, looking at figure 3.6. Qualitatively the results are very similar though. The deviations are very likely the result of differences in used numerical schemes. Since the qualitative results are the same, the model is assumed to be of

great value in investigating relations between adaptation of the heating function and the intensification of the tropical cyclone.

# Chapter 4

# Model results

## 4.1 Experiment descriptions

The numerical model is used to perform different experiments, in which the diabatic heating function $Q$ is altered in shape. For the experiments the heating function is represented by an exponential: $Q(r) = Q_0 e^{-(\frac{r-r_0}{a})^2}$. In this function three parameters can be changed: $Q_0$, the maximum value of the heating function, $r_0$, the radius at which the maximum value of the heating function occurs and $a$, the half-width of the heating function.

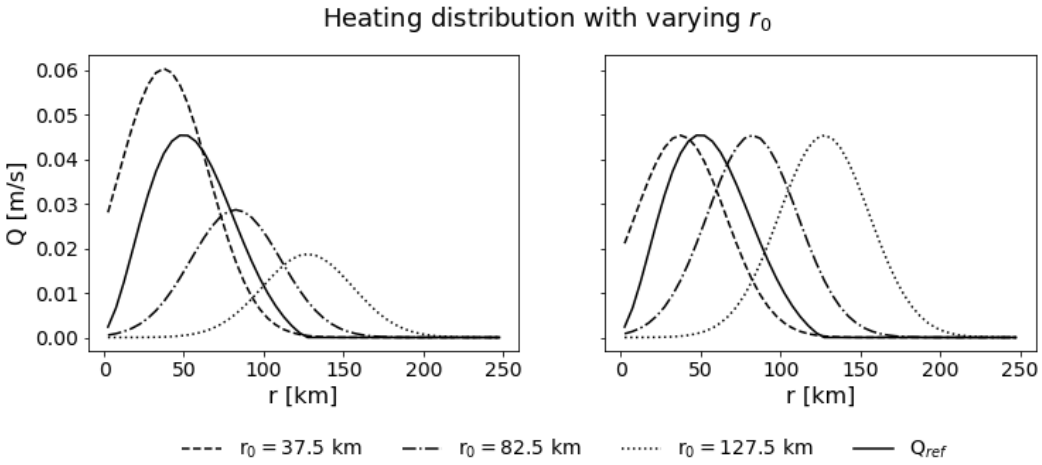


FIGURE 4.1: Heating distribution by Ekman pumping ($Q_{ref}$) and representation of the heating distribution by $Q(r) = Q_0 e^{-(\frac{r-r_0}{a})^2}$ for different values of $r_0$ and $a = 40$ km. The maximum heating $Q_0$ is such that the total potential energy is independent of $r_0$ (left) or is equal to the reference value (right)

The solid lines in figure 4.1 indicate the heating distribution $Q$ as generated by the Ekman pumping in the numerical model. The dashed and dotted lines are exponential representative functions in which the radius of maximum heating is altered while keeping the width constant and the maximum heating either constant or variable such that the potential energy change per unit time is kept constant. The change in potential energy per second for each radial grid point per unit density and per unit radian of arc due to the diabatic heating is determined by evaluating

$$\frac{\partial PE}{\partial t} = g(1-\epsilon)Qh_1 r dr. \tag{4.1}$$

We are interested in the increase of kinetic energy in the system that follows the increase in potential energy due to the diabatic heating. Part of the potential energy will be lost in the form of e.g. gravity-inertia waves, the remainder will contribute to the total kinetic energy of the primary and secondary flow. We will only consider the change in kinetic energy of the primary (swirling) circulation, which is the main criterion for determining the intensity of a tropical cyclone. The kinetic energy per unit density and per unit radian of arc due to the tangential flow for each radial grid point is given by

$$KE = \frac{1}{2}(h_1 r v_1^2 + \epsilon h_2 r v_2^2) dr \tag{4.2}$$

The increase in kinetic energy per unit time is evaluated by a forward difference, since the heating distribution at time step $n$ is a diagnostic function and thus only affects the balanced state at time $n+1$ . After this evaluation of the time rate of change of kinetic energy, energy conversion ratio $\frac{\Delta KE}{\Delta PE}$ of the diabatic heating can be calculated by dividing the time rate of change of kinetic energy by the time rate of change in potential energy.

A second quantity representing the intensification of a tropical cyclone is the central surface pressure tendency $\frac{\partial p_0}{\partial t}(r = 0)$. An axisymmetric vortex in gradient wind balance with a deepening central surface pressure by definition must coincide with increasing tangential wind speeds. To calculate the central $\frac{\partial p_0}{\partial t}$, we simply evaluate $\frac{\partial \phi_1}{\partial t}$ at $r = 0$. This is a reasonable approximation since $\rho$ is very close to unity and the pressure in the boundary layer is approximately constant.

The numerical tests start with initial conditions

$$v_1 = v_{max} \frac{2(r/r_{max})}{1 + (r/r_{max})^2} \tag{4.3a}$$

$$v_2 = 0 \tag{4.3b}$$

with $r_{max} = 50\,\text{km}$ and $v_{max}$ variable.

The numerical tests can be divided into three main categories, which are separate from each other based on the applied diabatic heating. The first category consists of all experiments starting with a free spinning vortex forced by a heating function with fixed $Q_0$ and variable $r_0$ and $a$. The second category includes the experiments initialized with a free spinning vortex forced by a heating function with variable $r_0$ and $a$ and $Q_0$ changing in such a way that $\Delta PE$ is constant. The third category consists of experiments where only the intensity of the diabatic heating is varied.

### 4.1.1 Constant $Q_0$

Figure 4.2 shows the results of the tests with constant maximum heating on a vortex initialized with $v_{max} = 10, 20, 30, 40$ m/s, from top to bottom. $Q_0$ is equal to the value found for the diabatic heating that results from boundary convergence in the numerical model, with $\eta = 2$. In the plots of the central surface pressure tendency, shown on the left, generally two distinct areas are visible. In each area, the tendency of the central surface pressure is either positive or negative. Negative values of the central surface pressure tendency, corresponding to deepening of the cyclone, are mainly found for heating distributions where $a$ is about half $r_0$ or more. A positive central surface pressure tendency occurs mainly when $a$ is less then half of $r_0$. So we may observe that deepening of the cyclone occurs mainly when the heating is broadly distributed and reaches close to the center of the cyclone, while filling occurs

when the heating is more narrowly distributed, and has a clearly distinguishable annular shape. However if we look more closely to the figure, we also observe that for all $a$'s, the central surface pressure tendency increases with decreasing $r_0$ for $r_0 < 25\,\text{km}$. This holds for all $v_{max}$'s except $v_{max} = 40\,\text{m/s}$. Therefore, for vortices with $v_{max} \leq 30\,\text{m/s}$, there is a minimum central surface pressure tendency at a finite $r_0$.

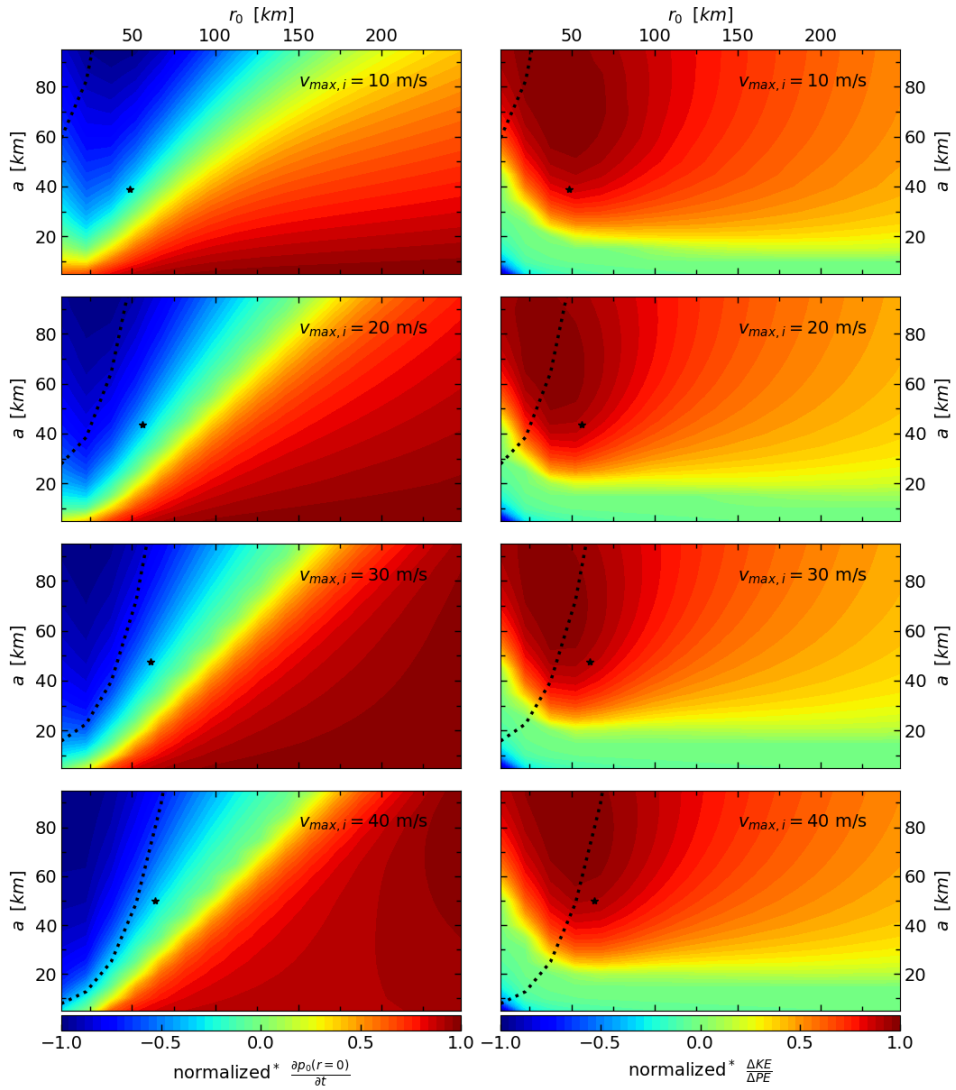


FIGURE 4.2: Central surface pressure tendency (left) and energy conversion ratio (right) for vortices initialized with $v_{max}$ ranging from 10 to 40 m/s (top to bottom). The internal Rossby deformation radius is represented by the dotted line as a function of radius (on the horizontal axis). The asterisk (*) denotes the $(a, r_0)$ coordinates corresponding to the diabatic heating that results from boundary layer convergence in the numerical model. $Q$ is varied with $Q_0$ held fixed. *The normalization is performed separately for all positive and negative values.

The right plots show the energy conversion ratio as a function of $r_0$ and $a$. This conversion ratio has a distinct maximum for a finite radius of maximum heating

$r_0$ and width $a$. The region where this maximum occurs is located in the same parameter space as where deepening may occur. The Rossby deformation radius in the center of the cyclone ranges from about 60 km for $v_{max} = 10\,\mathrm{m/s}$ to 9 km for $v_{max} = 40\,\mathrm{m/s}$.

The maximum of the energy conversion ratio for finite $r_0$ and $a$ is expected for a gradient wind adjustment event. As the size of the heating, in terms of width or radius, is too small, the energy of the perturbation by diabatic heating is converted completely into gravity-inertia waves. If the size of the heating is too large, the energy of the perturbation is converted for a relatively large amount into kinetic energy in the final balanced state. However the amount of conversion itself is limited. If the size of the heating compares well to the Rossby radius of deformation, both the amount energy being converted and the ratio of potential energy ending up in the kinetic energy of the balanced state over the lost energy are relatively high, leading to a maximum amount of intensification in terms of kinetic energy. The results for $v_{max} = 10, 20$ seem to support that $r_0$ in the order of $R_{cen}$ leads to the highest energy conversion ratio. The $a$ corresponding to the maximum energy conversion ratio is found to be in the same order of magnitude as the $R_{loc}$.

### 4.1.2 Constant $\Delta PE$

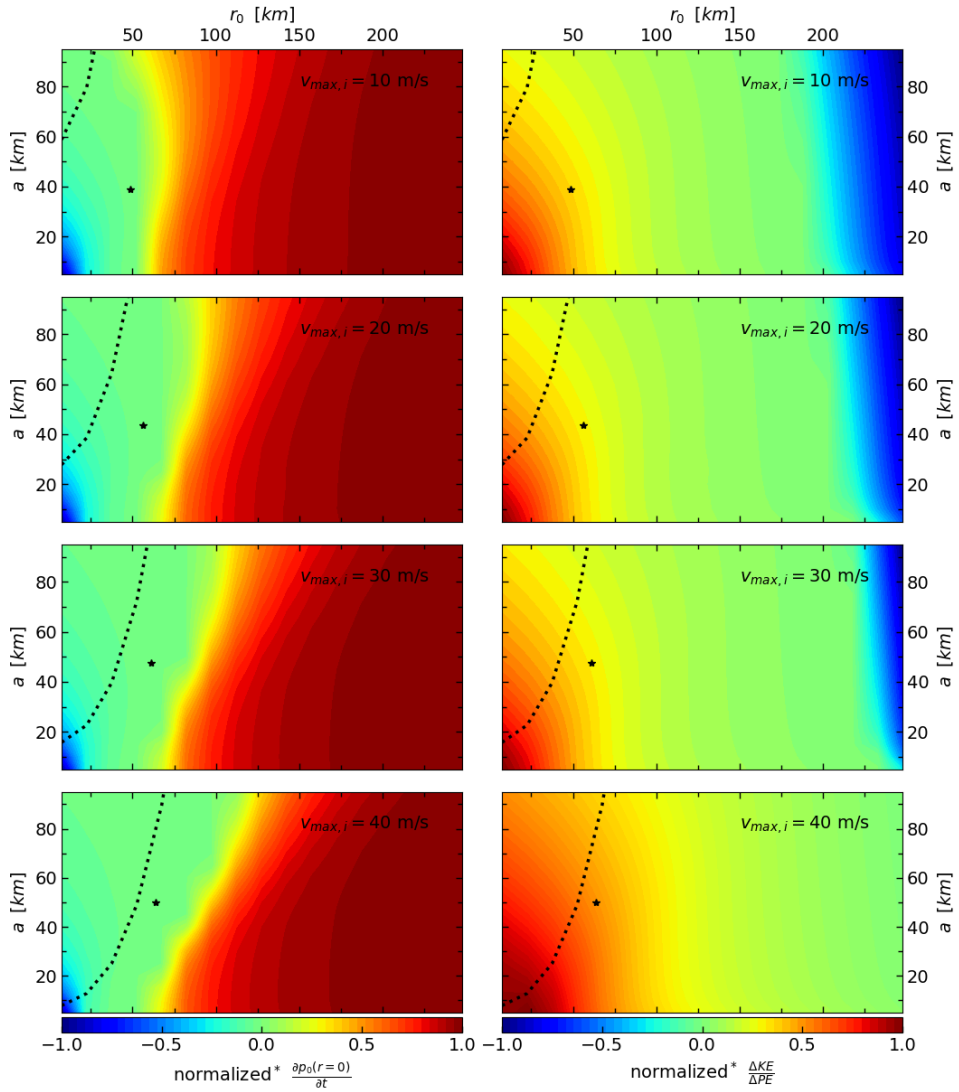


FIGURE 4.3: Central surface pressure tendency (left) and energy conversion ratio (right) for vortices initialized with $v_{max}$ ranging from 10 to 40 m/s (top to bottom). The internal Rossby deformation radius is represented by the dotted line as a function of radius (on the horizontal axis). The asterisk (*) denotes the $(a, r_0)$ coordinates corresponding to the diabatic heating that results from boundary layer convergence in the numerical model. $Q$ is varied with $\Delta PE$ held fixed. *The normalization is performed separately for all positive and negative values.

Figure 4.3 shows the normalized central surface pressure tendency and energy conversion ratio for four vortices initialized with maximum tangential wind speeds ranging from 10 to 40 m/s. The system is perturbed by a heating distribution whose $Q_0$, $r_0$ and $a$ vary in such a way that the potential energy added to the system ($\Delta PE$) is constant. Both for the central surface pressure tendency and energy conversion

ratio, it can be seen that the greatest amount of intensification is reached for distributions with small $r_0$ and $a$. Which corresponds to distributions that have very high intensity $Q_0$, due to the conservation of potential energy. Therefore, we will have a look at the results for changing $Q_0$ in the next subsection.

### 4.1.3 Changing $Q_0$

In figure 4.4 the central surface pressure tendency and energy conversion ratio are shown as function of intensity of the diabatic heating. The width and radius of the heating are equal to the width and radius of the heating distribution determined by frictional convergence and constant $\eta$. The central surface pressure tendency shows a linear decreasing behaviour with increasing intensity. The energy conversion ratio increases for small values of $Q_0 < 0.2$. Above this value, the effect of the intensity on the energy conversion ratio is very small.

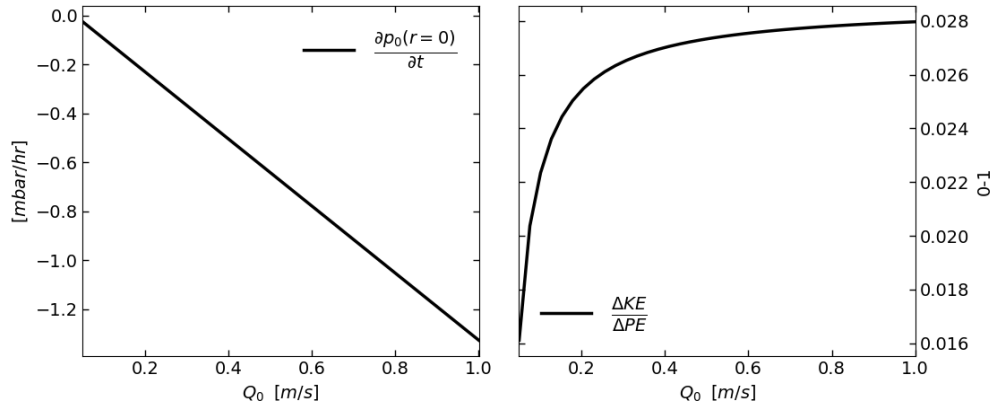


FIGURE 4.4: Results of varying the intensity $Q_0$ of heating in a vortex initialized with $v_{max} = 10\,\mathrm{ms}^{-1}$, and $a$ and $r_0$ equal to the values dictated by the convergence of the boundary layer in the normal run (Case A, Ooyama, 1969

## 4.2 Calculations

The numerical results of the previous section show the dependency of $\frac{\Delta KE}{\Delta PE}$ and $\frac{\partial p_0}{\partial t}$ on the parameters of the heating distribution. In figure 4.5 the results are shown as function of scale factor. The scale factors $a/R_{loc}$ and $r_0/R_{cen}$ are examined. The energy conversion ratio shows a maximum near a value of 1 for both scale factors with a maximum tangential wind speed of 10 m/s. As the vortex intensifies however, the maximum shifts towards higher values of $r_0/R_{cen}$. The fact that there is a maximum energy conversion ratio for both scale factors being close to one, is a result as expected for a vortex adjusting to gradient wind balance. If $r_0$ is much greater than $R_{cen}$, the heating can be seen as being dynamically too far away. This prohibits the conversion of potential energy into kinetic energy of the final balanced state. The central surface pressure tendency varies predominantly with $a/R_{loc}$, showing increased deepening with increasing $a/R_{loc}$.

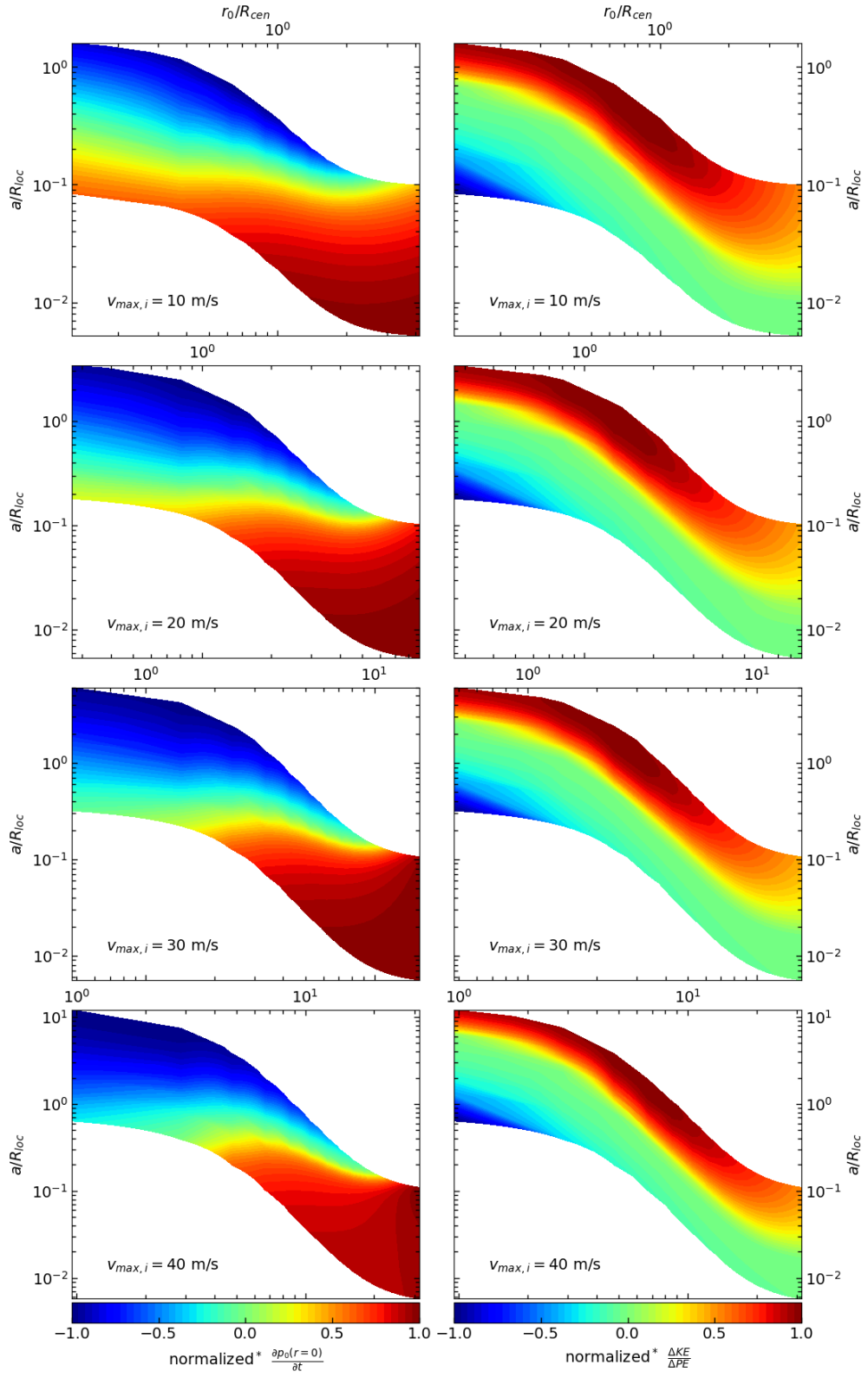


FIGURE 4.5: Central surface pressure tendency (left) and efficiency of the diabatic heating (right) for wind profiles with different maxima as function of the scale factors $a/R_{loc}$ and $r_0/R_{cen}$.

# Chapter 5

# Diagnostics of hurricane Irma (2017)

## 5.1 Synoptic overview

On the 30$^{th}$ of August 2017 00:00 UTC, a tropical depression formed over the Atlantic about 220 km west-southwest of São Vicente in the Cape Verde islands. While moving westwards, it rapidly gained strength in conditions of low vertical wind shear, a fairly moist lower troposphere and marginally high sea surface temperatures, according to Cangialosi, Latto, and Berg, 2018. After 6 hours the depression became a tropical storm, and just 24 hours later, 06:00 UTC 31 August, it became a hurricane with the name Irma. By 00:00 UTC 1 September Irma reached major hurricane status (wind speeds >= 185 km/h), 48 hours after genesis. The high intensification rate within this period is reached by roughly 1 in 30 Atlantic tropical cyclones. While the hurricane was very intense after this period, the inner core was rather compact with hurricane-force winds extending no more than about 28 km from the center.

From 00:00 UTC 1 September to 00:00 UTC 4 September Irma's intensification paused and she fluctuated between being a category 2 and 3 hurricane on the Saffir-Simpson scale. These fluctuations were likely due to eyewall replacements and intrusions of dry air. After this pause Irma started intensifying again. Irma reached a maximum intensity at 18:00 UTC September 5: 287 km/h tangential wind speeds and a central surface pressure of 914 hPa. By now Irma was a category 5 hurricane on the Saffir-Simpson scale. With this intensity, it made landfall on Barbuda at 05:45 UTC September 6. At 11:15 and 16:30 UTC Irma made landfall on St. Martin and Virgin Gorda in the Britis Virgin Islands, both with 287 km/h (category 5) tangential wind speeds. Later that day, as Irma was still a category 5 hurricane, reconnaissance aircraft measurements indicated that the hurricane had weakened and the tangential wind profile had a double maximum, indicative of concentric eyewalls. This was also supported by Doppler radar data from Puerto Rico.

At 05:00 UTC September 8 Irma made landfall on Little Inagua Island in the Bahamas with category 4 intensity, ending a 60 hr period of sustained category 5 intensity. This makes it the second longest sustained category 5 hurricane on record after the 1932 Cuba hurricane of Santa Cruz del Sur. Reconnaissance and radar data indicate that the core quickly became better organized and 18 hours after weakening below the intensity threshold, it again reached category 5. At 03:00 UTC September 9 Irma made its fifth landfall on Cuba. By 18:00 UTC that day, Irma had weakened due to interaction with land over the Cuban Keys down to a category 2 hurricane, before re-intensifying above the warm waters of the Florida Straits and making landfall on 13:00 UTC September 10 in Cudjoe Key as a category 4 hurricane. Irma made its seventh and final landfall near Marco Island, Florida, as a category 3 hurricane at

19:30 UTC that day. In total, Irma has caused 47 direct and 82 indirect deaths. The total damage is estimated to be around 50 billion USD.

## 5.2 Data preparation

Hurricane Irma is analyzed using a forecast from the non-hydrostatic HARMONIE model (HIRLAM ALADIN Research on Mesoscale Operational NWP in Euromed model), which is operational at the Koninklijk Nederlands Meteorologisch Instituut (KNMI) for short-range weather forecasts. The forecast data is provided on a Lambert Conformal horizontal grid with roughly 3.2 km resolution. The longitudes range from about 74 to 44 degrees West, the latitudes from roughly 6 to 24 degrees North. The model provides output on 16 pressure levels from 1000 to 50 hPa and the surface level, which is necessary for the calculation of the cross-isentropic mass flux. The forecast is initialized on 05-09-2017 at 12:00 UTC and lasts 36 hours, providing data on hourly intervals. The forecast period coincides with a period of growth while over warm waters and consequent period of constant maximum wind and minimum surface pressure. The data will be used to compare the findings of the numerical model with the forecast model. In order to do so, the data has to be pretreated. This process will be described in the following paragraphs.

### 5.2.1 Interpolation to isentropic levels

The data is provided on 16 pressure levels, and one surface level. The pressure levels are: 1000, 950, 925, 900, 850, 800, 700, 600, 500, 400, 300, 250, 200, 150, 100 and 50 hPa. The diabatic heating function in the shallow water model is a representation of cross-isentropic mass flow in an atmosphere divided into isentropic levels. Therefore the data needs to be interpolated from the pressure levels to the levels of constant potential temperature. This is done using the interpolation scheme provided by Edouard, Vautard, and Brunet, 1997. The potential temperature $\theta_k$ and value $f_k$ known on each pressure level with index $k$ (increasing with distance from the surface) are used to interpolate any variable $f$ to an isentropic level with potential temperature $\theta$.

$$f(\theta) = f_{k+1} - \frac{\ln \frac{\theta_{k+1}}{\theta}}{\ln \frac{\theta_{k+1}}{\theta_k}} (f_{k+1} - f_k) \tag{5.1}$$

Due to the approximate linear relation between $\ln p$ and $\ln \theta$ that exists in the atmosphere an interpolation of $\ln p$ is performed in the above way resulting in an interpolation for $p$ that is given by

$$p(\theta) = p_{k+1} \left( \frac{\theta}{\theta_{k+1}} \right)^{\gamma_k}, \quad \text{where } \gamma_k = \frac{\ln \frac{p_{k+1}}{p_k}}{\ln \frac{\theta_{k+1}}{\theta_k}} \tag{5.2}$$

The above methods are able to produce reliable interpolated results for all quantities except for the isentropic density $\sigma$, according to the authors. To calculate $\sigma$, first $\frac{\partial \theta}{\partial p}_k$ is estimated by fitting a parabola in $\ln \theta - \ln p$ coordinates through levels $k-1, k, k+1$ and taking the vertical derivative. This is done by evaluating

$$\left( \frac{\partial \theta}{\partial p} \right)_k = \frac{\theta_k}{p_k} \left( \frac{\ln \frac{p_k}{p_{k-1}} \ln \frac{\theta_{k+1}}{\theta_k}}{\ln \frac{p_{k+1}}{p_k} \ln \frac{p_{k+1}}{p_{k-1}}} + \frac{\ln \frac{p_{k+1}}{p_k} \ln \frac{\theta_k}{\theta_{k-1}}}{\ln \frac{p_k}{p_{k-1}} \ln \frac{p_{k+1}}{p_{k-1}}} \right). \tag{5.3}$$

Finally $\frac{\partial\theta}{\partial p}$ can be treated as any variable $f$ in equation 5.1 and is interpolated to isentropic levels. $\sigma$ simply follows from

$$\sigma = -\frac{\partial p}{g\partial\theta}. \tag{5.4}$$

The `Python` code performing the interpolation is shown in appendix B.1.

### 5.2.2 Cross-isentropic mass flux

The interpolated quantities are used to compute the cross-isentropic mass flux. The method that is applied in this research to do so followed from personal communication with dr. A.J. van Delden, and is explained here.

The horizontal velocities and pressure in isentropic coordinates are known on a grid of latitude/longitude coordinates. A quadrangle can be specified bounded by the latitudes $\phi_1$ and $\phi_2$ and the longitudes $\lambda_1$ and $\lambda_2$, where $\phi_2, \lambda_2 > \phi_1, \lambda_1$. A control surface within this quadrangle, vertically bounded by adjacent isentropic surfaces is used to determine the cross-isentropic mass flux. The control volumes have eight corner points where variables $u$, $v$, $p$ and $\theta$ are known. The mass budget is written down in equation 5.5.

$$T = (\overline{FX_1} - \overline{FX_2}) + (\overline{FY_1} - \overline{FY_2}) + (\overline{F\theta_1} - \overline{F\theta_2}) \tag{5.5}$$

In this equation, $T$ represents the time rate of change of mass within the control volume in units of $\mathrm{kgs}^{-1}$, $\overline{FX_1}$ and $\overline{FX_2}$ the zonal mass flux through the control volume boundary at latitudes $\phi_1$ and $\phi_2$, respectively. Likewise, $\overline{FY_1}$ and $\overline{FY_2}$ represent the meridional mass flux through the boundaries at longitudes $\lambda_1$ and $\lambda_2$, and $\overline{F\theta_1}$ and $\overline{F\theta_2}$ form the cross-isentropic mass flux at potential temperature levels $\theta_1$ and $\theta_2$. All mass fluxes are in units of $\mathrm{kgs}^{-1}$

Assuming hydrostatic balance, the horizontally averaged vertical pressure difference between the lower and upper bounding isentropic surface $\Delta p = p_{\theta_1} - p_{\theta_2}$ is equal to the weight of the air inside the control volume per unit horizontal area. Therefore the mass tendency can be written as

$$T = \frac{S}{g}\frac{\Delta p_{n+1} - \Delta p_n}{\Delta t}, \tag{5.6}$$

where $S$ represents the horizontal area covered by the control volume in $\mathrm{m}^2$, $n$ represents the time index and $\Delta t$ the time difference between consecutive time steps in s.

The mass flux through the vertical bounding surfaces at longitudes $\lambda_1$ and $\lambda_2$ are the product of the zonal mass flux $\frac{\Delta\bar{p}}{g}\bar{u}$ and the meridional distance $ad\phi = a(\phi_2 - \phi_1)$, where $a$ is the radius of the earth. The horizontal bar indicates meridional averaging. The zonal mass flux at the bounding surfaces can be written down as

$$\overline{FX_i} = \frac{\Delta\bar{p}}{g}\bar{u}ad\phi. \tag{5.7}$$

Likewise, the meridional mass flux $FY$ through the vertical bounding surfaces at $\phi_1$ and $\phi_2$ are the product of the meridional mass flux $\frac{\Delta\bar{p}}{g}\bar{v}$ and the zonal distance $a\cos(\phi)d\lambda$. So we have:

$$\overline{FY_i} = \frac{\Delta\bar{p}}{g}\bar{v}a\cos(\phi)d\lambda \tag{5.8}$$

In the lower troposphere, isentropic surfaces might intersect with the surface of the earth leading to missing data where the isentropes are below the surface. To correctly estimate the cross-isentropic mass flux, the control volumes closest to the surface are bounded by the ground level grid points, instead of isentropic level grid points. There is no further difference in the calculations of the mass fluxes. The vertical mass flux at the surface of the earth is zero, therefore the flux difference $\overline{F\theta_1} - \overline{F\theta_2} = -\overline{F\theta_2}$ between the first isentropic level above ground and ground level gives us the vertical mass flux at the first isentropic level. In the same way, the vertical mass flux can be evaluated for all higher levels by adding the flux differences of all levels. The cross-isentropic mass flux $\overline{CIMF}$ is obtained by dividing by the surface area $S$,

$$\overline{CIMF}_i \equiv \overline{\left(\sigma \frac{\partial \theta}{\partial t}\right)}_i = \frac{\overline{F\theta_i}}{S}, \tag{5.9}$$

where $i$ indicates the index of the isentropic level and $\sigma$ the isentropic density, which is defined by

$$\sigma = -\frac{1}{g}\frac{\partial p}{\partial \theta} \tag{5.10}$$

The `Python` code performing the calculation of the cros-isentropic mass flux is shown in appendix B.2. An example of the results of the cross-isentropic mass flux calculations is shown in figure 5.1. The example shows clear band structures, circulating around an eye with a clearly distinguishable eyewall, where much of the diabatic heating takes place. In the eye there is a strong diabatic cooling, related to vaporization and melting of water particles.

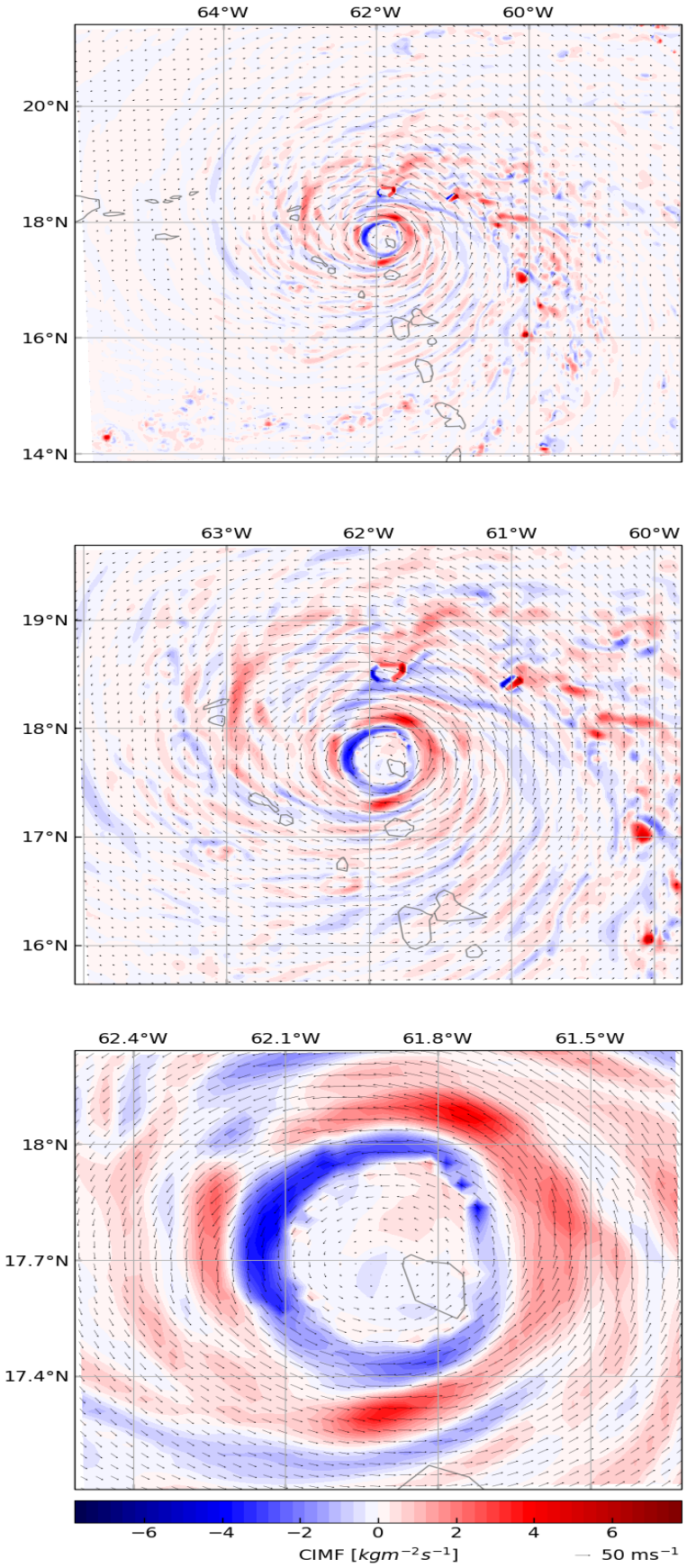


FIGURE 5.1: Snapshot of the cross-isentropic mass flux at the 340K isentrope at T+20 hr in hurricane Irma.

### 5.2.3 Azimuthal averaging

The final step before the data of hurricane Irma is compared to the results of the numerical model is to take azimuthal averages around the center of the hurricane. The location of the center is determined on each isentropic level by fitting a parabola through the grid point of minimum velocity and the adjacent grid points in the zonal and meridional direction. For the isentropic levels in the lower troposphere, where the velocity field forms a typical vortical structure, the center is easily defined in this way. For isentropic levels in the higher troposphere and lower levels which coincide with the earth's surface, the center is not that easily recognized. Therefore central latitude and longitude values are compared with the average of the levels 320 - 350 K. Deviations of more than 0.08 °N/E are replaced by this average. Using the central latitude and longitude, a cylindrical grid is constructed, and the velocity field is decomposed into a radial and tangential field. Using the fast `numpy.histogram()` method, the data is averaged per bin of 5 km radial interval. In this way the data is azimuthally averaged for all levels and time steps. The `Python` code responsible for the azimuthal averaging can be found in appendix B.3.

## 5.3 Data analysis

The radial and vertical distributions of several azimuthally averaged variables are shown in figure 5.2. The top right subplot shows the cross-isentropic mass flux and radial velocity. In this subplot we can see the typical structure of the overturning circulation. In the boundary layer, there is a strong inflow. Close to the center, in the eyewall, the moist air rises and releases large amounts of latent heat. At the 360 K isentrope, the air parcels flow outwards to larger radii. In the eyewall, just above the boundary layer, there is a region of outflow. This can be explained by the fact that parcels in the top part of the boundary layer are drawn towards the center of the cyclone without being affected too much by the friction of the earth's surface, such that they approximately conserve angular momentum. This creates supergradient winds in the boundary layer near the vortex core. As the air parcels rise up in the eyewall and transcend the boundary layer, there is no such inflow of air to maintain their supergradient tangential velocities and they move to larger radii where they will obtain gradient tangential velocities. This explains the region of outflow in the eyewall just above the boundary layer. High in the troposphere, close to the 380 K isentrope inside the eye, there is a region of radial inflow. This inflow feeds the radial outflow in the eye at roughly 370 K. In the descending air column inside the eye, there is negative cross-isentropic flow. This is the result from melting and vaporization of water particles.

Although the central pressure in the core of a tropical cyclone is relatively low compared to the ambient pressure at the same height, the isobars in the top right subplot show that there is an increase of pressure at isentropic levels near the vortex center. Because of the warm core, isentropic surfaces tend to bend towards the earth's surface near the center, hence the increased pressure. Around the center, the tangential velocity is largely determined by the gradient wind balance. The pressure in the eye decreases less fast with height than the pressure outside the eye, due to the warm core. The radial pressure gradient therefore decreases with height, hence explaining the decreasing tangential velocity with height. The maximum tangential velocity in vortex core is largely determined by spin up in the boundary layer, while the tangential velocity outside the eyewall is determined by spin up of the air above the boundary layer. The latter occurs with a slow inflow above the boundary layer

caused by the diabatic heating. Note that although this inflow might not be as fast as the inflow in the boundary layer, it does not have to be to cause spin up as there is virtually no loss of angular momentum above the boundary layer. Both spin up processes are related because the deep convection, which is related to the boundary layer convergence, also entrains tropospheric air above the boundary layer.

The left plot on the second row shows the radial and vertical distribution of the Brunt-Väisälä frequency, $N = \sqrt{\frac{g}{\theta}\frac{\partial\theta}{\partial z}}$, a measure of the static stability against vertical displacement of an air parcel. In the outflow layer, $N$ is relatively low. This is a consequence of the organized convective motion within the tropical cyclone. As air picks up (mainly) latent heat near the earth's surface and releases it during the approximate moist adiabatic ascent, the regions where convection takes place warm up. This lowers the vertical potential temperature gradient in the outflow layer. Hence $N$ is relatively low in the outflow layer. In the vortex core, a dipole structure of $N$ is present due to the warm core and corresponding vertical potential temperature gradient.

The right plot on the second row shows the absolute vorticity, $\zeta_a = \zeta + f$, and absolute angular momentum per unit mass, $M = vr + 0.5fr^2$. The absolute vorticity is positive nearly everywhere. In the outflow layer, however, the absolute vorticity is small and at some locations becomes negative, indicating low inertial stability, or even inertial instability. It is in this region, where one might expect the tangential circulation to be anti-cyclonic. As the air parcels high in the outflow layer tend to originate from low absolute angular momentum surfaces, the effect of the earth's rotation is enough to reverse the sign of the tangential wind. Spin up of the vortex can be seen as a radially inward movement of the absolute angular momentum surfaces.

The left plot on the third row shows the vertical velocity in isentropic coordinates and the Exner function $\Pi = c_p\left(\frac{p}{p_{ref}}\right)^{2/7}$ contours. The vertical velocity shows roughly the same structure as the cross-isentropic mass flux. Above the region of downwelling inside the eye, the negative vertical velocity is very large. In this area, the static stability is relatively low. Therefore, this negative vertical velocity is more easily accomplished. The maximum vertical velocity is about 30 K/hr. This is likely a reasonable value, as Bui et al., 2009 found vertical velocities of 21 K/hr for a weaker vortex ($v_{max} \approx 50$ m/s) using a three-dimensional non-hydrostatic model.

The right plot on the third row shows the distribution of potential vorticity $PV = \frac{f+\zeta}{\sigma}$ and isentropic density $\sigma = \frac{-\partial p}{g\partial\theta}$. In the vortex center, there is a positive anomaly of potential vorticity, extending from the earth's surface to a height of about 360 K, with a maximum roughly at the 340 K isentrope. The high anomaly is due to a combination of high absolute vorticity and low isentropic density within this region. Directly above the potential vorticity anomaly, the isentropes bend down towards the positive anomaly, creating a positive isentropic density anomaly.

The bottom left plot shows the distribution of $I = \sqrt{(f+\zeta)(f+2\frac{v}{r})}$, which is a measure of the inertial (centrifugal) stability against radial displacement. Near the eyewall, this stability is the highest, while in the outflow region there is little inertial stability. The missing values in the graph are related to the locally negative values of absolute vorticity. Contours depict the vertical height above sea level, from which the bending down of the isentropes in the vortex core can be seen.

The bottom right plot shows the local Rossby radius of deformation. It is defined as the ratio $NH/I$. So the horizontal size scales with the ratio $N/I$ given the scale height $H$. The effect of high inertial stability is clearly visible by very small Rossby radii, up to about 50 km in the eyewall. Because of the low absolute vorticity values

at larger radii the Rossby radius quickly grows to very high values with increasing radius.

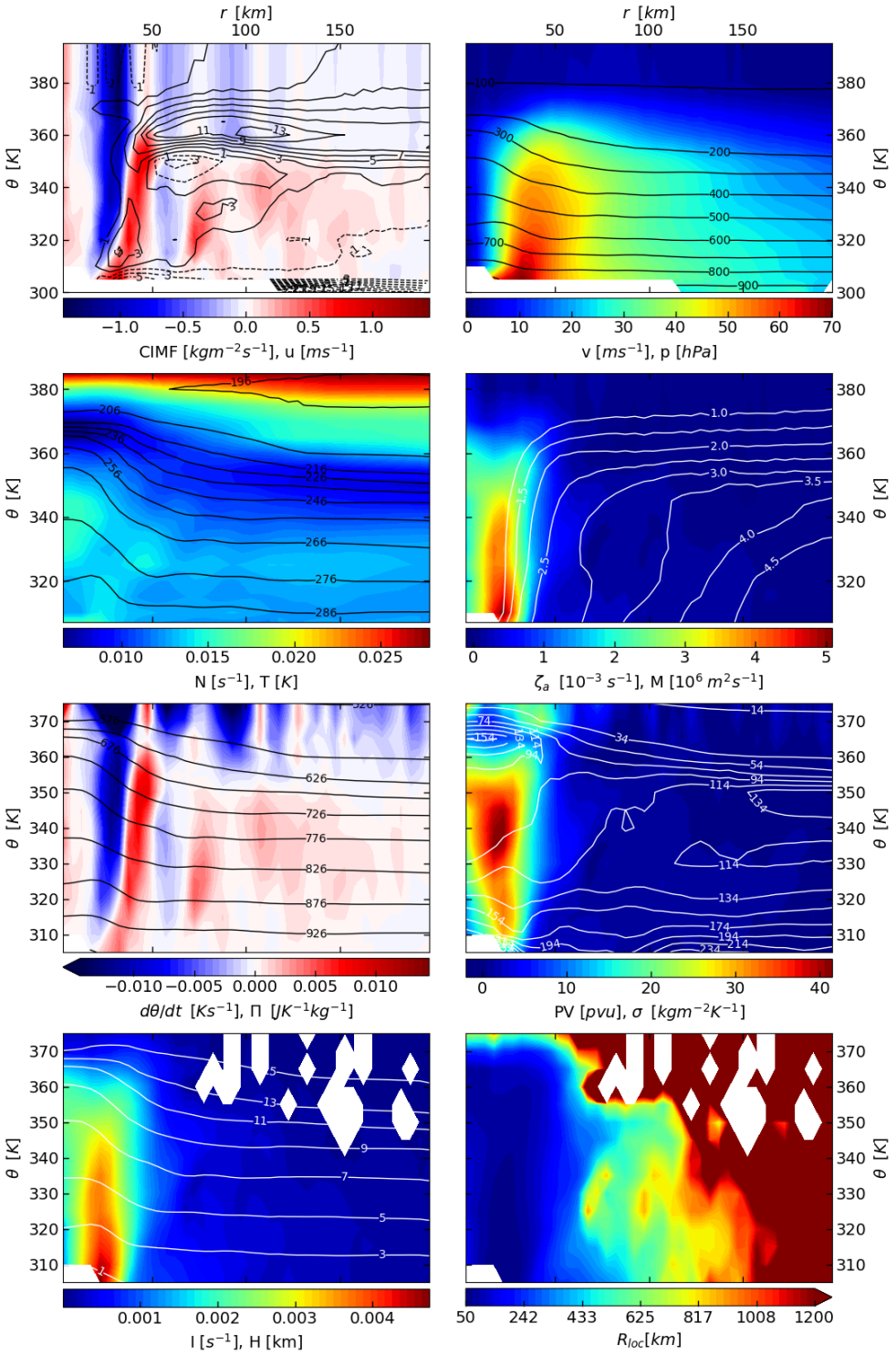


FIGURE 5.2: Azimuthally averaged quantities from the forecast run of Irma at T+20 hr as function of radius, $r$, and potential temperature, $\theta$. Where two variables are mentioned below the diagrams, the first variable is depicted using filled contours, the second using contour lines. Dashed contour lines represent negative values.

Several variables as a function of time are shown in figure 5.3. The numerical computation of the scale of the heating, accompanied by some result plots, is shown in appendix B.4. Of interest for this research is the relation between the intensification of the cyclone and the scale of the diabatic heating. Therefore the radius of maximum heating, $r_0$ and width of the heating, $2a$, are plotted with the Rossby radius at $r = 0$, $R_{cen}$, and the Rossby radius evaluated at $r = r_0$, $R_{loc}$. $R_{loc}$ is higher than $R_{cen}$ due to the low inertial stability, $I$, at $r = r_0$ compared to $r = 0$. In the same figure, the energy conversion ratio, intensity of the heating, the total kinetic energy and central surface pressure are shown. We see that the vortex is in a deepening stage, where the central surface pressure decreases by roughly 20 hPa in about 30 hour. The solid and dashed lines in the top left plot show $r_0$ and $R_{cen}$. In the period of 10-15 hours after the initial time, there is an increase of $R_{cen}$, while $r_0$ remains approximately constant. During the same period, the central surface pressure slightly increases. The energy conversion ratio remains approximately constant while $R_{loc}$ and $a$ remain fairly constant over time. $Q_0$ is shown on the same plot as the energy conversion ratio. There does not seem to be an observable relation between the central pressure and $Q_0$, or between the energy conversion ratio and $Q_0$.

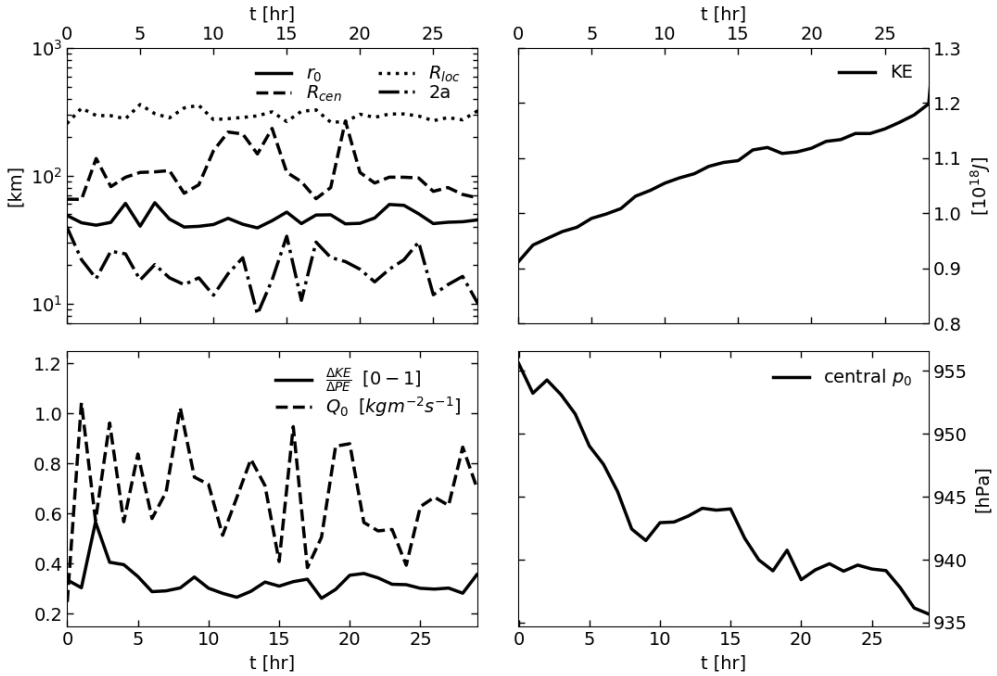


FIGURE 5.3: Time evolution of Irma for a comparison between the Rossby radius of deformation and the scale of the diabatic heating, and the central surface pressure and kinetic energy.

# Chapter 6

# Conclusion and discussion

## 6.1 Conclusion

In this research we have investigated the relation between the intensification of a tropical cyclone, in terms of central surface pressure and the efficiency of diabatic heating, and the width of the diabatic heating, radius of the diabatic heating and the intensity of the diabatic heating. Numerical experiments with the two-layer model of Ooyama showed that the efficiency of the diabatic heating, defined as the ratio of kinetic energy gained in the balanced state by the potential energy added in the form of a diabatic mass flux, is greatest when the radius of maximum heating is comparable to the central Rossby radius, and the width of the heating is comparable to the local Rossby radius. Deepening of the tropical cyclone in terms of central surface pressure seems to be favourable for distributions with widths comparable to, or greater than the local Rossby radius. Changing the intensity of the diabatic heating in the numerical model greatly affects the energy conversion ratio for vertical mass fluxes (per unit density) smaller than $0.2\,\mathrm{ms}^{-1}$, but does not show much influence for stronger diabatic heating. The central surface pressure tendency is found to decrease linearly with increasing intensity of the diabatic heating.

A data analysis has been performed using forecast data of hurricane Irma. The data has been generated by the high-resolution non-hydrostatic HARMONIE model. A calculation of the cross-isentropic mass flux in hurricane Irma has provided insight in the three dimensional structure of the overturning circulation. Several plots of azimuthally averaged variables have been presented to form a conceptual understanding of the tropical cyclone.

In order to assess the relation between the intensification and shape of the diabatic heating, we compared the time evolution of the radius of maximum heating, width of the heating, intensity of the heating, the local and central Rossby radius, the energy conversion ratio, the kinetic energy and central surface pressure. It has been found that the central surface pressure, during the general deepening stage, increased for a few hours while the central Rossby radius was anomalously high during the same period. The energy conversion efficiency is relatively constant over time, as is the increase of kinetic energy. The fluctuations in the intensity of the diabatic heating presumably have no effect on the intensification of the tropical cyclone during the forecast period.

The results from the 2-layer model seem to support the idea that the efficiency of the diabatic heating in tropical cyclone intensification can be interpreted as a process of gradient wind adjustment, preferring scales of the heating associated with the Rossby radius of deformation. The data analysis results provide no strong evidence of gradient wind adjustment being the main driver of tropical cyclone intensification, yet do not contradict the theory either.

## 6.2 Discussion

The results presented in this study should be interpreted with care. The numerical tests with Ooyama's model were performed using simple, bulk-aerodynamic approximations. Solutions might depend critically on approximations for heat and momentum fluxes in the boundary layer. The time evolution of hurricane Irma is hard to interpret as a process of gradient wind adjustment, as there is only a limited number of data points in time. It might be tempting to assign the evolution of the model to changes in the shape of the diabatic heating, while other factors, such as the environmental wind shear, might play an important role.

## Appendix A

# Numerical model

```
import numpy as np

def initialize(v_max=10,r_max=50000,n=None):
    """Initialize all variables"""
    _h[:,:,:] = 0
    v[:,:,:] = 0
    _phi[:,:,:] = 0
    psi[:,:,:] = 0
    psib[:,:] = 0
    _Q[:,:] = 0
    _w[:,:] = 0
    _eta[:,:] = 0
    _u[:,:,:] = 0
    u0[:,:] = 0
    _chi0[:,:] = 0
    _chi2[:,:] = 0
    _chis[:,:] = 0
    S[:,:,:] = 0
    KE[:,:] = 0
    PE[:,:] = 0
    _QP[:,:] = 0
    eff[:] = 0
    dpdt[:] = 0
    Rdef[:,:] = 0

    #Initialize tangential wind
    if n: # Load from previous run
        loc = 'C:/Users/jsprd/Desktop/Afstuderen/model_data'
        data = np.load(loc+'/regular_v_eta_t.npy').item()
        v[:,:,0] = data['v'][:,:K,n]
        _eta[:,0] = data['eta'][:K-1,n]
    else: # Make new profile
        v[0,:,0] = v_max * 2 * (r/r_max) / (1 + (r/r_max)**2)
        v[1,:,0] = 0
        _eta[:,0] = ETA

    # Initialize geopotential
    with np.errstate(invalid='ignore'):
        dphidr = np.nan_to_num((f + v[:,:,0]/r) * v[:,:,0])
```

```
    _phi[:,:,0] = np.cumsum(dphidr[:,:-1]*dr, axis=1)
    _phi[0,:,0] -= _phi[0,-1,0]
    _phi[1,:,0] -= _phi[1,-1,0]

    # Initialize all chis
    _chi2[:,0] = alpha * cp**-1 * \
        (_phi[1,:,0] - _phi[0,:,0]) + _CHI2
    _chi0[:,0] = (_eta[:,0] - 1) * \
        (_chi2[:,0] - _chi1) + _chi2[:,0]
    _chis[:,0] = -beta * cp**-1 * _phi[0,:,0] + _CHIs

def weighted_average(x):
    """Compute weighted average of x"""
    msg = 'x has wrong shape %s'%(x.shape,)
    assert x.shape in [(J,K),(J,K-1),(K,),(K-1,)],msg

    if x.shape[0] == J:
        wa1 = weighted_average(x[0,:])
        wa2 = weighted_average(x[1,:])
        return np.array([wa1, wa2])

    if len(x) == K-1:
        wa = np.zeros((K))
        _xr = x * _r
        wa[1:-1] = (_xr[1:] + _xr[:-1]) / (2 * r[1:-1])
        wa[0] = 3 * (wa[1] - wa[2]) + wa[3]
        wa[-1] = 3 * (wa[-2] - wa[-3]) + wa[-4]
        return wa
    elif len(x) == K:
        xr = x * r
        return (xr[1:] + xr[:-1]) / (2 * _r)
    else:
        print("warning: x has length ",len(x))
        return x

def ddr(x):
    """Compute central radial derivative of x"""
    if x.shape.count(K) == 1:
        dim = x.shape.index(K)
    elif x.shape.count(K-1) == 1:
        dim = x.shape.index(K-1)
    else:
        msg = 'Cant differentiate x with shape %s'%(x.shape,)
        raise AssertionError(msg)

    dxdr = np.diff(x, axis=dim) / dr

    return dxdr

def temp_extrapolate(x,n):
    """Linearly extrapolate x to time step n"""
```

```
    assert n>1,'n must be at least 2'
    msg = 'x has wrong shape: %s'%(x.shape,)
    assert x.shape in [(J,K,N),(J,K-1,N)], msg
    y = 2 * x[:,:,n-1] - x[:,:,n-2]
    return y

def calc_B(v,F,_G):
    """Compute B (forcing radial flow equation)"""
    assert v.shape == (J,K),'v has wrong shape: %s'%(v.shape,)
    assert F.shape == (J,K),'F has wrong shape: %s'%(F.shape,)
    assert _G.shape == (J,K-1), \
        '_G has wrong shape: %s'%(_G.shape,)

    # Calculate dGdr (mass distribution term)
    d_Gdr = np.zeros((J,K))
    d_Gdr[:,1:-1] = ddr(_G)
    d_Gdr[:,-1] = 2 * d_Gdr[:,-2] - d_Gdr[:,-3]

    # Calculate B
    B = np.zeros((J,K))
    B[:,1:] = 1/g*((f+2*v[:,1:]/r[1:])*F[:,1:] - \
        r[1:]*d_Gdr[:,1:])
    B[:,0] = 0

    return B

def calc_F(_h,_Q,v):
    """Compute F (angular momentum redistribution term)"""
    assert _h.shape == (J,K-1), \
        '_h has wrong shape: %s'%(_h.shape,)
    assert _Q.shape == (K-1,), \
        '_Q has wrong shape: %s'%(_Q.shape,)
    assert v.shape == (J,K),'v has wrong shape: %s'%(v.shape,)

    # Calculate lateral eddy flux
    v_r = np.zeros(v.shape)
    v_r[:,1:] = v[:,1:] / r[1:]
    v_r[:,0] = 2 * v_r[:,1] - v_r[:,2]

    _L = np.array([
        lambda1*_h[0,:]*_r**3 * ddr(v_r[0,:]),
        eps*lambda2*_h[1,:]*_r**3 * ddr(v_r[1,:])
    ])

    # Translate from staggered to regular grid
    dLdr_r = np.zeros((v.shape))
    dLdr_r[:,1:-1] = ddr(_L) / r[1:-1]
    dLdr_r[:,0] = 0
    dLdr_r[:,-1] = 2 * dLdr_r[:,-2] - dLdr_r[:,-3]
    h = weighted_average(_h)
    Q = weighted_average(_Q)
```

```
    Qpos = Q.clip(min=0)
    Qneg = -Q.clip(max=0)

    # Calculate F
    F = np.array([
        1/h[0,:]*( (Qneg + mu)*(v[1,:] - v[0,:])*r \
            + dLdr_r[0,:] ),
        1/(eps*h[1,:])*( (Qpos + mu)*(v[0,:] - v[1,:])*r \
            + dLdr_r[1,:] )
    ])

    assert F[0,0] == 0 and F[1,0] == 0, \
        'F does not vanish at r=0'
    return F

def calc_G(_w,_Q):
    """Compute G (mass redistribution term)"""
    assert _w.shape == (K-1,), \
        '_w has wrong shape: %s'%(_w.shape,)
    assert _Q.shape == (K-1,), \
        '_Q has wrong shape: %s'%(_Q.shape,)

    _G = np.array([
        g * _w,
        g * (_w + (1-eps)/eps * _Q)
    ])

    return _G

def dvdt(v,_h,psi,F):
    """Prognostic solution for v"""
    assert v.shape == (J,K),'v has wrong shape: %s'%(v.shape,)
    assert _h.shape == (J,K-1), \
        '_h has wrong shape: %s'%(_h.shape,)
    assert psi.shape == (J,K), \
        'psi has wrong shape: %s'%(psi.shape,)
    assert F.shape == (J,K),'F has wrong shape: %s'%(F.shape,)

    # Calculate relative vorticity (upstream difference)
    vr = np.c_[[0,0], v*r, 2*v[:,-1]*r[-1] - v[:,-2]*r[-2]]
    diff = (psi>=0) * (vr[:,2:]-vr[:,1:-1]) \
        + (psi<0) * (vr[:,1:-1]-vr[:,:-2])
    with np.errstate(divide='ignore',invalid='ignore'):
        zeta = diff / (r * dr)
        zeta[:,0] = 2 * zeta[:,1] - zeta[:,2]

    # Transformation to regular grid
    h = weighted_average(_h)

    # Calculate dvdt
    dvrdt = np.array([
```

```
        (f+zeta[0,:])/h[0,:] * psi[0,:] + F[0,:],
        (f+zeta[1,:])/(eps*h[1,:]) * psi[1,:] + F[1,:]
    ])

    with np.errstate(divide='ignore',invalid='ignore'):
        dvdt = dvrdt / r
        dvdt[:,0] = 0

    return dvdt

def solve_psi(v0,v1,_h0,B0,psi,psib):
    """Calculate the radial mass flux"""
    assert v0.shape == (J,K), \
        'v0 has wrong shape: %s'%(v0.shape,)
    assert v1.shape == (J,K), \
        'v1 has wrong shape: %s'%(v1.shape,)
    assert _h0.shape == (J,K-1), \
        '_h0 has wrong shape: %s'%(_h0.shape,)
    assert B0.shape == (J,K), \
        'B0 has wrong shape: %s'%(B0.shape,)
    assert psi.shape == (J,K), \
        'psi has wrong shape: %s'%(psi.shape,)
    assert psib.shape == (K,), \
        'psib has wrong shape: %s'%(psib.shape,)

    # Calc. rel. vorticity (upstream split level difference)
    if np.max(np.abs((v1-v0)))<1e-12:
        vr = np.c_[[0,0], v0*r, \
            2*v0[:,-1]*r[-1]-v0[:,-2]*r[-1]]
        diff = (psi>=0) * (vr[:,2:]-vr[:,1:-1]) \
            + (psi<0) * (vr[:,1:-1]-vr[:,:-2])
    else:
        v0r = np.c_[[0,0], v0*r, \
            2*v0[:,-1]*r[-1]-v0[:,-2]*r[-1]]
        v1r = np.c_[[0,0], v1*r, \
            2*v1[:,-1]*r[-1]-v1[:,-2]*r[-1]]
        diff = ((psi>=0) * ( (v0r[:,2:]-v0r[:,1:-1]) \
            + (v1r[:,1:-1]-v1r[:,:-2]) ) \
            + (psi<0) * ( (v1r[:,2:]-v1r[:,1:-1]) \
            + (v0r[:,1:-1]-v0r[:,:-2]) )) / 2
    with np.errstate(divide='ignore',invalid='ignore'):
        zeta = diff / (r * dr)
        zeta[:,0] = 2 * zeta[:,1] - zeta[:,2]

    # Formulate linear system matrix
    h0 = weighted_average(_h0)
    with np.errstate(divide='ignore',invalid='ignore'):
        v_r = v0 / r
        v_r[:,0] = 2 * v_r[:,1] - v_r[:,2]
        S = (f+2*v_r)*(f+zeta)/(g*h0)
    a = np.arange(K)/(np.arange(K)+0.5)
```

```
    c = np.arange(K)/(np.arange(K)-0.5)
    b = a + c + dr**2/(1-eps)*S
    M = np.zeros((J+1,K,K))
    for i in range(1,K-1):
        M[:,i,i+1] = a[i]
        M[:2,i,i] = -b[:,i]
        M[2,i,i] = a[i] + c[i]
        M[:,i,i-1] = c[i]
    M[:2,0,0] = 1
    M[:2,-1,-1] = 1
    M[1,-1,-2] = - (2*R-dr)/(2*R+dr)

    def loop(psi):
        V = np.array([
            dr**2*(B0[0,:] - eps*B0[1,:] \
                - S[1,:]*psi[1,:])/(1 - eps),
            eps*dr**2*(B0[1,:] - B0[0,:] \
                - S[0,:]*psi[0,:])/(1 - eps)
        ])
        V[:,0] = 0
        V[0,-1] = -(psi[1,-1]+psib[-1])
        V[1,-1] = 0
        psi_new = np.array([
            np.linalg.solve(M[0,:,:],V[0,:]),
            np.linalg.solve(M[1,:,:],V[1,:])
        ])
        psi_new[:,0] = 0
        return psi_new

    dif = 1
    count = 0
    while dif > 1e-5:
        psi_new = loop(psi)
        dif = np.max(np.abs(psi_new-psi))
        psi[:,:] = psi_new
        count += 1
        if count > 250:
            print('    Iteration failed',end='\n')
            break

    assert (psi_new[1,-1] - psi_new[1,-2])/dr \
        + (psi_new[1,-1] + psi_new[1,-2])/2/R < 1e-5
    assert psi_new[0,-1] + psi_new[1,-1] + psib[-1] < 1e-5
    return psi_new,S

def dphidt(psi,_G):
    """Prognostic solution for the geopotential"""
    assert psi.shape == (J,K), \
        'psi has wrong shape: %s'%(psi.shape,)
    assert _G.shape == (J,K-1), \
        '_G has wrong shape: %s'%(_G.shape,)
```

```
    Psi = np.array([
        psi[0,:] + psi[1,:],
        psi[0,:] + eps**-1 * psi[1,:]
    ])

    _dphidt = g * ddr(Psi)/_r + _G

    return _dphidt

def gradientwind(_phi):
    """Return the gradient wind velocity"""
    assert _phi.shape == (J,K-1), \
        'phi has wrong shape: %s'%(_phi.shape,)

    v = np.zeros((J,K))
    D = (f*r[1:-1])**2 + 4*r[1:-1]*ddr(_phi)
    if np.min(D)<0:
        print("    warning: no solution for v",end='\n')
        return v,D
    v[:,1:-1] = (-f*r[1:-1] + np.sqrt(D)) / 2
    v[:,0] = 0
    v[:,-1] = 2 * v[:,-2] - v[:,-3]
    return v,D

def calc_h(_phi):
    """Calculate thickness"""
    assert _phi.shape == (J,K-1), \
        'phi has wrong shape: %s'%(_phi.shape,)

    _h = np.array([
        h1 + (_phi[0,:] - eps*_phi[1,:]) / (g * (1 - eps)),
        h2 + (_phi[1,:] - _phi[0,:]) / (g * (1 - eps))
    ])

    return _h

def calc_psib(Cd,v):
    """Calculate boundary layer radial mass flux"""
    assert v.shape == (J,K),'v has wrong shape: %s'%(v.shape,)

    vr = v[0,:]*r
    zeta = weighted_average( ddr(vr)/_r )

    psib = Cd*v[0,:]*np.abs(v[0,:])*r/(f+zeta)

    return psib

def calc_w(psi0):
    """Calc. vertical velocity air at top of boundary layer"""
    assert psi0.shape == (K,), \
```

```
        'psi0 has wrong shape: %s'%(psi0.shape,)

    _w = ddr(psi0)/_r

    return _w

def calc_Q(_eta,_w,_h,fixed_param = None):
    """Return vertical mass flux due to diabatic heating"""
    assert _w.shape == (K-1,), \
        'w has wrong shape: %s'%(_w.shape,)

    if fixed_param:
        rmax = fixed_param[0]
        width = fixed_param[1]
        if fixed_param[2] is None:
            _Qref = _eta * _w
            _Qref[_Qref<0] = 0
            _QP = g * (1-eps) * np.sum(_Qref*_h[1,:]*_r*dr)
            Q0 = _QP / (g*(1-eps) \
                *np.sum(np.exp(-((_r-rmax)/width)**2) \
                *_h[1,:]*_r*dr))
        else:
            Q0 = fixed_param[2]
        _Q = Q0 * np.exp(-((_r-rmax)/width)**2)
    else:
        _Q = _eta * _w
        _Q[_Q<0] = 0

    return _Q

def calc_chi0(_chi0,_chis,psib,_w,v1,dt,Ce):
    """Calculate boundary layer chi"""
    assert psib.shape == (K,2), \
        'psib has wrong shape: %s'%(psib.shape,)
    assert _chis.shape == (K-1,2), \
        '_chis has wrong shape: %s'%(_chis.shape,)
    assert _chi0.shape == (K-1,), \
        '_chi0 has wrong shape: %s'%(_chi0.shape,)
    assert _w.shape == (K-1,2), \
        '_w has wrong shape: %s'%(_w.shape,)
    assert v1.shape == (K,2), \
        'v1 has wrong shape: %s'%(v1.shape,)
    assert Ce.shape == (K,), \
        'Ce has wrong shape: %s'%(Ce.shape,)

    # Calculate amount of needed iterations
    with np.errstate(invalid='ignore',divide='ignore'):
        u0 = -psib[:,0] / (h0 * r)
        u1 = -psib[:,1] / (h0 * r)
        u0[0] = 0
        u1[0] = 0
```

```
        _u0 = weighted_average(u0)
        _u1 = weighted_average(u1)
        umax = np.max(np.abs(_u0))
    nr_steps = int(np.ceil(dt / (0.4 * dr / umax)))

    # Useful variables
    dt0 = dt / nr_steps
    _v1a = weighted_average(np.abs(v1[:,0]))
    _v1na = weighted_average(np.abs(v1[:,1]))
    _wneg = 0.5 * (np.abs(_w[:,0]) - _w[:,0])
    _wnneg = 0.5 * (np.abs(_w[:,1]) - _w[:,1])
    _Ce = weighted_average(Ce)

    def dchi0dt(_chi0,f_wneg,f_v1a,f_chis,f_u):
        chi0 = np.concatenate(([2*_chi0[0]-_chi0[1]], \
            _chi0,[2*_chi0[-1]-_chi0[-2]]))
        dchi0 = (f_u<0) * (chi0[2:]-chi0[1:-1]) \
            + (f_u>=0) * (chi0[1:-1]-chi0[:-2])
        udchi0dr = f_u * dchi0 / dr
        return -udchi0dr - f_wneg/h0*(_chi0-_chi1) \
            + _Ce*f_v1a/h0*(f_chis-_chi0)

    # Iterate for _chi0
    for n in range(nr_steps):
        f_u = _u0 + n/nr_steps * (_u1 - _u0)
        f_wneg = _wneg + n/nr_steps * (_wnneg - _wneg)
        f_v1a = _v1a + n/nr_steps * (_v1na - _v1a)
        f_chis = _chis[:,0] + n/nr_steps \
            * (_chis[:,1] - _chis[:,0])
        _chi0 += dchi0dt(_chi0,f_wneg,f_v1a,f_chis,f_u) * dt0

    return _chi0

def calc_K(_h,v):
    """Return the total kinetic energy
    per radial grid interval dr"""
    h = weighted_average(_h)
    K1 = 0.5 * h[0,:] * r * v[0,:]**2 * dr
    K2 = 0.5 * eps * h[1,:] * r * v[1,:]**2 * dr
    K = K1+K2
    return K

def calc_P(_h):
    """Return total potential energy anomaly
    per radial grid interval dr"""
    h = weighted_average(_h)
    P0 = g/2 * ((1-eps) * h1**2 + eps * (h1+h2)**2) * r*dr
    P = g/2 * ((1-eps) * h[0,:]**2  \
        + eps*(h[0,:] + h[1,:])**2 ) * r*dr
    return P-P0
```

```
def calc_QP(_Q,_h):
    """Return energy of heating used to increase
    potential energy per dr"""
    return g*(1-eps)*_Q*_h[1,:]*_r*dr

def calc_eff(KE1,KE0,_QP,dt):
    """Return total energy conversion ratio"""
    dKEdt = (KE1 - KE0) / dt
    eff = np.sum(dKEdt) / np.sum(_QP)
    return eff

def rossby_radius(_h,v):
    """Calculate rossby radius of internal deformation"""
    _zeta = 1/_r * ddr(v[0,:]*r)
    _R = np.sqrt((1-eps) * g * _h[0,:] / 2) * 1/(f + _zeta)
    return _R

def run(Q_param=None,printprogress=True):
    """Run model for given parameters"""
    dt = float(3600) # [s] dummy dt
    n_adjust = 0
    for n in range(N):
        _h[:,:,n] = calc_h(_phi[:,:,n])
        Rdef[:,n] = rossby_radius(_h[:,:,n],v[:,:,n])
        KE[:,n] = calc_K(_h[:,:,n],v[:,:,n])
        PE[:,n] = calc_P(_h[:,:,n])
        Cd = (0.5+0.06*v[0,:,n])*1e-3
        psib[:,n] = calc_psib(Cd,v[:,:,n])
        with np.errstate(invalid='ignore',divide='ignore'):
            u0[:,n] = -psib[:,n]/(h0*r)
            u0[0,n] = 0
        _w[:,n] = calc_w(psib[:,n])
        _chis[:,n] = -beta * cp**-1 * _phi[0,:,n] + _CHIs
        if n > 0:
            _chi0[:,n] = calc_chi0(_chi0[:,n-1], \
                _chis[:,n-1:n+1], psib[:,n-1:n+1], \
                _w[:,n-1:n+1], v[0,:,n-1:n+1], dt, Cd)
        _chi2[:,n] = alpha * cp**-1 \
            * (_phi[1,:,n] - _phi[0,:,n]) + _CHI2
        _eta[:,n] = 1 + (_chi0[:,n]-_chi2[:,n]) \
            /(_chi2[:,n]-_chi1)
        _Q[:,n] = calc_Q(_eta[:,n],_w[:,n],_h[:,:,n], \
            fixed_param=Q_param)
        _QP[:,n] = calc_QP(_Q[:,n],_h[:,:,n])
        F = calc_F(_h[:,:,n],_Q[:,n],v[:,:,n])
        _G = calc_G(_w[:,n],_Q[:,n])
        B = calc_B(v[:,:,n],F,_G)
        if n - n_adjust >= 2:
            psi_ = temp_extrapolate(psi,n)
        else:
            psi_,S_ = solve_psi(v[:,:,n],v[:,:,n], \
```

```
                _h[:,:,n],B,psi[:,:,n],psib[:,n])
        u_ = weighted_average(psi_) / (_h[:,:,n] * _r[None,:])
        umax = np.max(np.abs(u_))
        delta = umax*dt/dr
        if delta**2 > 0.5:
            dt = 0.4 * dr / umax
            n_adjustdt = n
        elif delta**2 < 0.4:
            dt = 0.5 * dr / umax
            n_adjustdt = n
        else:
            pass
        dv = dvdt(v[:,:,n],_h[:,:,n],psi_,F) * dt
        psi[:,:,n],S[:,:,n] = solve_psi(v[:,:,n], \
            v[:,:,n]+dv,_h[:,:,n],B,psi_,psib[:,n])
        _u[:,:,n] = weighted_average(psi[:,:,n]) \
            / (_h[:,:,n] * _r)
        _dphi = dphidt(psi[:,:,n],_G) * dt
        if n > 0:
            eff[n-1] = calc_eff(KE[:,n],KE[:,n-1], \
                _QP[:,n-1],t[n]-t[n-1])
            dpdt[n-1] = (_phi[0,0,n] - _phi[0,0,n-1]) \
                / (t[n] - t[n-1])

        if n+1 < N:
            t[n+1] = t[n] + dt
            _phi[:,:,n+1] = _phi[:,:,n] + _dphi
            v[:,:,n+1],D = gradientwind(_phi[:,:,n+1])

        if printprogress:
            print('\rProgress: %0.1f%% (n=%i)'% \
                ((n+1)/N*100,n),end='')

# Constants
eps = float(0.9)    # Hydrostatic stability parameter
h0 = float(1e3)     # Depth boundary layer [m]
h1= float(5e3)      # Depth bottom layer [m]
h2= float(5e3)      # Depth top layer [m]
g = float(9.81)     # Gravitational constant [m/s^2]
f= float(5e-5)      # Coriolis parameter [s^-1]
R = float(1e6)      # Rossby radius [m]
_CHIs = float(30)   # Initial surface chi [K]
_CHI0 = float(10)   # Initial boundary layer chi [K]
_chi1 = float(-10)  # Initial bottom layer chi [K]
_CHI2 = float(0)    # Initial top layer chi [K]
alpha = float(10)   # Coefficient chi_2 vs phi2-phi1
beta = float(2)     # Coefficient p dependency Chi_s
lambda1=float(1e3)  # Kinematic eddy viscosity [m^2/s]
lambda2=float(1e3)  # Kinematic eddy viscosity [m^2/s]
mu = float(5e-4)    # Coeff. of tang. shearing stress [m/s]
v_max= float(10)    # Initial maximum tangential wind [m/s]
```

```
r_max= float(5e4)  # Radius of maximum tangential wind [m]
ETA = float(2)     # Initial value entrainment parameter
cp = float(1006)   # Sp. heat air at constant pressure [J/kgK]

# Model parameters
J=int(2)                          # Number of main layers
K=int(200)                        # Number of radial grid points
N=int(1100)                       # Number of time steps
dr= float(5000)                   # Radial grid size [m]
r = np.arange(K)*dr               # Radial grid [m]
_r = (np.arange(K-1)+0.5)*dr # Staggered radial grid [m]
t = np.zeros(N)                   # Time grid [s]

# Variables
_u = np.zeros((J,K-1,N))  # Radial velocity [m/s]
u0 = np.zeros((K,N))      # Rad. velocity boundary layer [m/s]
v = np.zeros((J,K,N))     # Tangential wind [m/s]
_phi = np.zeros((J,K-1,N))# Dev. from geopotential [m^2/s^2]
_h = np.zeros((J,K-1,N))  # Layer depth [m]
_Q = np.zeros((K-1,N))    # Diabatic mass flux [m/s]
_w = np.zeros((K-1,N))    # Mass flux top bound. layer [m/s]
_eta = np.zeros((K-1,N))  # Entrainment parameter
_chi0 = np.zeros((K-1,N)) # Chi boundary layer
_chi2 = np.zeros((K-1,N)) # Chi top layer
_chis = np.zeros((K-1,N)) # Chi surface
psi = np.zeros((J,K,N))   # Radial mass flux [m^3/s]
psib = np.zeros((K,N))    # Boundary layer radial flux [m^3/s]
S = np.zeros((J,K,N))     # Inertial stability [m^-2]
KE = np.zeros((K,N))      # Kinetic energy per dr [J/m]
PE = np.zeros((K,N))      # Potential energy per dr [J/m]
_QP = np.zeros((K-1,N))   # Converted heating energy [J/s]
eff = np.zeros((N))       # Energy conversion ratio
dpdt = np.zeros((N))      # Central surf. p. tendency [hPa/hr]
Rdef = np.zeros((K-1,N))  # Rossby radius [m]

initialize()
run()
```

# Appendix B

# Irma

## B.1 Interpolation scheme

```
# Import modules
import numpy as np
import glob
import os

# Set i/o-paths
readpath = 'F:/afstuderen/npyFilesIrmaIsobaric/'
writepath = 'F:/afstuderen/npyFilesIrmaIsentropic/'
name = 'fc2017090512'
outfile = writepath + name + '_isentropic'
filenames = [os.path.basename(x) for x in \
    glob.glob(readpath+name+'_isobaric+***.npy')]
outfile = writepath + name + '_isentropic'

# Constants
g = 9.81 # gravitational constant [m/s**2]
p_ref = 1000 #reference pressure [hPa]
kappa = 2/7 # R/c_p [J/kg/K]

# Calculate potential temperature and isentropic
# density on pressure levels
# and store in existing .npy files
for fid,fname in enumerate(filenames):

    print('\rCalculating isentropic density for %s'%fname, \
        end='')
    ds = np.load(readpath+filenames[fid]).item()

    theta = ds['plt'] * (p_ref/ds['lvl'] \
        [:,None,None])**kappa #potential temperature
    t0 = theta[:-2,:,:]
    t1 = theta[1:-1,:,:]
    t2 = theta[2:,:,:]
    p0 = 100 * ds['lvl'][:-2,None,None]
    p1 = 100 * ds['lvl'][1:-1,None,None]
    p2 = 100 * ds['lvl'][2:,None,None]
```

```
    dtdp = t1/p1 * (np.log(p1/p0)*np.log(t2/t1) \
        / (np.log(p2/p1)*np.log(p2/p0)) \
        + np.log(p2/p1)*np.log(t1/t0) \
        / (np.log(p1/p0)*np.log(p2/p0)) )

    sigma_p = -1/g * dtdp**-1

    ds['pt'] = theta
    ds['sig'] = sigma_p

    np.save(readpath+filenames[fid], ds)

# Interpolate u,v,p,sigma to given isentropic levels
def interp2isentropes(lvls_pt):

    M = len(lvls_pt)
    (J,K,L) = ds['pt'].shape
    pres_pt = np.zeros((M,K,L))
    uvel_pt = np.zeros((M,K,L))
    vvel_pt = np.zeros((M,K,L))
    sigma_pt = np.zeros((M,K,L))

    # Determine index of pressure level directly below
    # theta levels for all grid points and theta levels
    idcs = np.zeros((M,K,L),dtype=np.int)
    for pt_id,pt in enumerate(lvls_pt):
        found = np.zeros((K,L),dtype=np.bool)
        for p in range(0,J-1):
            c1 = (ds['pt'][p+1,:,:] > pt)
            c2 = (ds['pt'][p,:,:] < pt)
            crossed_isentrope = (c1 & c2) & ~found
            idcs[pt_id,:,:] += crossed_isentrope * p
            found[crossed_isentrope] = True


    # Pick data from adjacent pressure levels
    # (0: above (lower pressure), 1:below,
    # 2:second below, 3...)
    t0 = np.choose(idcs-1, ds['pt'], mode='clip')
    t1 = np.choose(idcs, ds['pt'], mode='clip')
    t2 = np.choose(idcs+1, ds['pt'], mode='clip')
    t3 = np.choose(idcs+2, ds['pt'], mode='clip')
    u1 = np.choose(idcs, ds['plu'], mode='clip')
    u2 = np.choose(idcs+1, ds['plu'], mode='clip')
    v1 = np.choose(idcs, ds['plv'], mode='clip')
    v2 = np.choose(idcs+1, ds['plv'], mode='clip')
    pres_p = ds['lvl'][:,None,None] * np.ones((J,K,L))
    p0 = np.choose(idcs-1, pres_p, mode='clip')
    p1 = np.choose(idcs, pres_p, mode='clip')
    p2 = np.choose(idcs+1, pres_p, mode='clip')
    p3 = np.choose(idcs+2, pres_p, mode='clip')
```

```
    t0[idcs==0] = np.nan
    t1[idcs==0] = np.nan
    t2[idcs==0] = np.nan
    t3[idcs==0] = np.nan
    u1[idcs==0] = np.nan
    u2[idcs==0] = np.nan
    v1[idcs==0] = np.nan
    v2[idcs==0] = np.nan
    p0[idcs==0] = np.nan
    p1[idcs==0] = np.nan
    p3[idcs==0] = np.nan
    p2[idcs==0] = np.nan

    with np.errstate(invalid='ignore'):
        uvel_pt = u2 - np.log(t2/lvls_pt[:,None,None]) \
            / np.log(t2/t1) * (u2 - u1)
        vvel_pt = v2 - np.log(t2/lvls_pt[:,None,None]) \
            / np.log(t2/t1) * (v2 - v1)
        gamma = np.log(p2/p1) / np.log(t2/t1)
        pres_pt = p2 * (lvls_pt[:,None,None]/t2)**gamma
        dtdp1 = 0.01*t1/p1*(np.log(p1/p0)*np.log(t2/t1)) \
            / (np.log(p2/p1)*np.log(p2/p0)) \
            + 0.01*t1/p1*(np.log(p2/p1)*np.log(t1/t0)) \
            / (np.log(p1/p0)*np.log(p2/p0))
        dtdp2 = 0.01*t2/p2*(np.log(p2/p1)*np.log(t3/t2)) \
            / (np.log(p3/p2)*np.log(p3/p1)) \
            + 0.01*t2/p2*(np.log(p3/p2)*np.log(t2/t1)) \
            / (np.log(p2/p1)*np.log(p3/p1))
        dtdp = dtdp2 - np.log(t2/lvls_pt[:,None,None]) \
            / np.log(t2/t1) * (dtdp2 - dtdp1)
        sigma_pt = -1/g * dtdp**-1

    return uvel_pt, vvel_pt, pres_pt, sigma_pt

# Execute interpolation for every time step,
# save in new .npy files
lvls = np.arange(300,400,5)
for f_id,fname in enumerate(filenames):
    print('\rInterpolating file %s [%i / %i]' \
        %(fname,f_id+1,len(filenames)),end='')
    ds = np.load(readpath+fname).item()
    u,v,p,sigma = interp2isentropes(lvls)
    irma_data = {
        'glp':ds['glp'],
        'tlu':u,
        'tlv':v,
        'tlp':p,
        'sig':sigma,
        'lvl':lvls,
        'lat':ds['lat'],
```

```
        'lon':ds['lon'],
        'tim':ds['tim']
    }
    np.save(outfile+'+%03i.npy'%f_id,irma_data)
```

## B.2 Calculate cross-isentropic mass flux

```
# Import modules
import numpy as np
import glob
import os

# Constants
g = 9.81      # gravitational constant [m/s^2]
a = 6371000   # radius earth [m]

# Set i/o-paths
path = 'F:/afstuderen/npyFilesIrmaIsentropic/'
name = 'fc2017090512'
filenames = [os.path.basename(x) for x in \
    glob.glob(path+name+'_isentropic+***.npy')]

# Calculate pressure difference isentropes
# with isentropic or ground level surface below
def calc_dp(tlp,glp):
    dp = np.zeros(tlp.shape)
    check_nan = True
    for pt in range(tlp.shape[0]):
        if pt == 0:
            dp[pt,:,:] = 100*tlp[pt,:,:] - glp[:,:]
        else:
            dp[pt,:,:] = 100*(tlp[pt,:,:] - tlp[pt-1,:,:])
            if check_nan:
                nan_idcs = np.isnan(dp[pt,:,:])
                dp[pt,:,:][nan_idcs] = 100* \
                    tlp[pt,:,:][nan_idcs] - glp[nan_idcs]
                if nan_idcs.any() == False:
                    check_nan = False
        with np.errstate(invalid='ignore'):
            dp[pt,:,:][dp[pt,:,:]>0] = np.nan

    return dp

# Calculate zonal and meridional mass flux
# within isentropic surfaces
def calc_massflux(tlu,tlv,dp):
    # Zonal mass flux
    dlat = latsr[1:,:] - latsr[:-1,:]
    # meridionally averaged zonal velocity
```

```
    ui = (tlu[:,1:,:] + tlu[:,:-1,:])/2
    # meridionally averaged pressure difference
    dpi = np.abs(dp[:,1:,:] + dp[:,:-1,:])/2
    FX = 1/g * dpi * ui * a * dlat[None,:,:]

    #Meridional mass flux
    dlon = lonsr[:,1:] - lonsr[:,:-1]
    # meridional velocity zonally averaged
    vi = (tlv[:,:,1:] + tlv[:,:,:-1])/2
    # pressure difference zonally averaged
    dpi = np.abs(dp[:,:,1:] + dp[:,:,:-1])/2
    coslat = np.cos((latsr[:,1:]+latsr[:,:-1])/2)
    FY = 1/g*dpi*vi*a * coslat[None,:,:] * dlon[None,:,:]
    return FX, FY

# Calculate local mass tendency between isentropic surfaces
def calc_masstendency(dp0,dp1,dt):

    #Mass tendency
    lonsr_ma = (lonsr[1:,:] + lonsr[:-1,:]) / 2
    latsr_za = (latsr[:,1:] + latsr[:,:-1]) / 2
    dlon = lonsr_ma[:,1:] - lonsr_ma[:,:-1]
    A = a**2 * (np.sin(latsr_za[1:,:]) - \
        np.sin(latsr_za[:-1,:]))* dlon #area
    dpi0 = np.abs(dp0[:,:-1,1:] + dp0[:,:-1,:-1] \
        + dp0[:,1:,1:] + dp0[:,1:,:-1])/4
    dpi1 = np.abs(dp1[:,:-1,1:] + dp1[:,:-1,:-1]  \
        + dp1[:,1:,1:] + dp1[:,1:,:-1])/4
    T = A[None,:,:] / g * (dpi1[:,:,:] - dpi0[:,:,:]) / dt
    return T, A

# Calculate Cross-Isentropic Mass Flux needed to account
# for the mass tendency
def calc_CIMF(FX,FY,T):
    FT = np.zeros((T.shape[0]+1,T.shape[1],T.shape[2]))
    CIMF = np.zeros(T.shape)
    dFX = FX[:,:,:-1] - FX[:,:,1:]
    dFY = FY[:,:-1,:] - FY[:,1:,:]
    check = True
    for pt in range(CIMF.shape[0]):
        FT[pt+1,:,:] = FT[pt,:,:] + dFX[pt,:,:] \
            +  dFY[pt,:,:] - T[pt,:,:]
        if check:
            nan_idcs = np.isnan(FT[pt+1,:,:])
            FT[pt+1,:,:][nan_idcs] = (dFX[pt,:,:] \
                +  dFY[pt,:,:] - T[pt,:,:])[nan_idcs]
            if nan_idcs.any() == False:
                check = False
        CIMF[pt,:,:] = FT[pt+1,:,:] / A[None,:,:]

    return CIMF
```

```
#Calculate CIMF for all time steps
ds0 = np.load(path+filenames[0]).item()

for f_id,fname in enumerate(filenames[:-1]):

    print('\rCalculating␣cross-isentropic␣mass␣flux␣for␣%s␣\
␣␣␣␣␣␣␣␣[%i␣/␣%i]'%(fname,f_id+1,len(filenames)-1), end='')
    ds1 = np.load(path+filenames[f_id+1]).item()

    # Pressure difference
    dp0 = calc_dp(ds0['tlp'],ds0['glp'])
    dp1 = calc_dp(ds1['tlp'],ds1['glp'])

    latsr = ds0['lat'] * np.pi/180
    lonsr = ds0['lon'] * np.pi/180

    # Zonal/Meridional mass flux
    FX, FY = calc_massflux(ds0['tlu'],ds0['tlv'],dp0)

    # Mass tendency
    dt = (ds1['tim']-ds0['tim']).item()
    T, A = calc_masstendency(dp0,dp1,dt)

    #calculate CIMF
    CIMF = calc_CIMF(FX,FY,T)
    ds0['CIMF'] = CIMF

    # Store CIMF
    np.save(path+fname, ds0)

    ds0 = ds1
```

## B.3 Azimuthal averaging

```
# Import modules
import glob
import os
import numpy as np

# Set i/o-paths
readpath = 'F:/afstuderen/npyFilesIrmaIsentropic/'
writepath = 'F:/afstuderen/npyFilesIrmaAveraged/'
name = 'fc2017090512'
filenames = [os.path.basename(x) for x in \
    glob.glob(readpath+name+'_isentropic+***.npy')]

# Determine center of cyclone
def findCenter(u,v):
```

```
    # Find location of maximum wind
    V = np.sqrt(u[100:-100,100:-100]**2 \
        + v[100:-100,100:-100]**2)
    try:
        y_maxv,x_maxv = np.unravel_index(np.argmax(V),V.shape)
    except ValueError:
        return 0,0

    # Search within 20 gridpoints for the minimum wind
    V_box = V[y_maxv-20:y_maxv+20,x_maxv-20:x_maxv+20]
    if (np.isnan(V_box).any() or V_box.size==0):
        return 0,0
    yc,xc = np.unravel_index(np.argmin(V_box),V_box.shape)
    yc += y_maxv - 20
    xc += x_maxv - 20

    # Fit parabola for better accuracy
    x = [-2,-1,0,1,2]
    y1 = V[yc-2:yc+3,xc]
    y2 = V[yc,xc-2:xc+3]
    a1,b1,c1 = np.polyfit(x,y1,deg=2)
    a2,b2,c2 = np.polyfit(x,y2,deg=2)
    yc += -b1/(2*a1)
    xc += -b2/(2*a2)

    yc += 100
    xc += 100

    xc = xc.clip(min=0,max=ds['lon'].shape[1]-1)
    yc = yc.clip(min=0,max=ds['lon'].shape[0]-1)

    return yc, xc

# Convert indices to latitude/longitude
def index2LatLon(x,y):

    lat1 = ds['lat'][int(np.floor(y)),int(x)]
    lat2 = ds['lat'][int(np.ceil(y)),int(x)]
    lon1 = ds['lon'][int(y),int(np.floor(x))]
    lon2 = ds['lon'][int(y),int(np.ceil(x))]
    latc = lat1 + (y % 1) * (lat2 - lat1)
    lonc = lon1 + (x % 1) * (lon2 - lon1)

    return latc, lonc

# Calculate cylindrical grid (radius, angle)
def calcRT(latc,lonc,r):
    dx = (ds['lon'] - lonc) * np.pi / 180 * a \
        * np.cos(ds['lat']/180*np.pi)
    dy = (ds['lat'] - latc)*np.pi/180*a
```

```
    r_grid = np.sqrt(dx**2 + dy**2)
    r_max = np.max(r_grid)

    with np.errstate(invalid='ignore'):
        t_grid = np.arctan(dy/dx)
        t_grid[dx<0] += np.pi
        t_grid[(dx>0)*(dy<0)] += 2*np.pi

    return r_grid, t_grid

# Calculate mean value of variable within bins
# (for radial interval)
def calc_hist(x,y,bins):
    x = x[~np.isnan(y)]
    y = y[~np.isnan(y)]
    n,_ = np.histogram(x,bins)
    sy,_ = np.histogram(x,bins,weights=y)

    with np.errstate(invalid='ignore'):
        mean = sy/n

    return mean,_

# Average u,v,p,sigma,cimf for each time step over radial
# intervals around cyclone center and store in .npy files
r = np.arange(0,700000,5000)
a = 6371000
f_id0 = 0

ds = np.load(readpath+filenames[0]).item()

latc = np.empty(len(ds['lvl']))
lonc = np.empty(len(ds['lvl']))
urad = np.empty((len(ds['lvl']),len(r)-1))
vtan = np.empty((len(ds['lvl']),len(r)-1))
tlp = np.empty((len(ds['lvl']),len(r)-1))
glp = np.empty(len(r)-1)
sig = np.empty((len(ds['lvl']),len(r)-1))
cimf = np.empty((len(ds['lvl']),len(r)-1))

for f_id in range(f_id0,len(filenames)):
    print('\rReading file %s'%filenames[f_id],end='')
    ds = np.load(readpath+filenames[f_id]).item()

    # Calculate central latitude/longitude for
    # all isentropic levels
    for lvl in range(len(ds['lvl'])):
        u = ds['tlu'][lvl,:,:]
        v = ds['tlv'][lvl,:,:]
        yc, xc = findCenter(u, v)
        latc[lvl], lonc[lvl] = index2LatLon(xc, yc)
```

```
    # If found location deviates from other levels [4-10],
    # replace by average of these levels
    latavg = np.sum(latc[4:10]) / 6
    lonavg = np.sum(lonc[4:10]) / 6
    latc[:][abs(latc[:]-latavg)>0.08] = latavg
    lonc[:][abs(lonc[:]-lonavg)>0.08] = lonavg

    # For each isentropic level, decompose velocity in
    # radial/tangential parts, calculate
    # averages for all quantities within specified radial
    # intervals around the cyclone center
    # and store in new .npy file
    for lvl in range(len(ds['lvl'])):
        r_grid, t_grid = calcRT(latc[lvl],lonc[lvl],r)
        rc_grid = (r_grid[1:,1:] + r_grid[:-1,:-1] + \
            r_grid[1:,:-1] + r_grid[:-1,1:])/4
        u_rad = ds['tlu'][lvl,:,:]*np.cos(t_grid[:,:]) + \
            ds['tlv'][lvl,:,:]*np.sin(t_grid[:,:])
        v_tan = -ds['tlu'][lvl,:,:]*np.sin(t_grid[:,:]) + \
            ds['tlv'][lvl,:,:]*np.cos(t_grid[:,:])
        idcs_sort = np.argsort(r_grid.ravel())
        idcsc_sort = np.argsort(rc_grid.ravel())
        r_sort = r_grid.ravel()[idcs_sort]
        rc_sort = rc_grid.ravel()[idcsc_sort]
        u_sort = u_rad.ravel()[idcs_sort]
        v_sort = v_tan.ravel()[idcs_sort]
        p_sort = ds['tlp'][lvl,:,:].ravel()[idcs_sort]
        glp_sort = ds['glp'][:,:].ravel()[idcs_sort]
        sig_sort = ds['sig'][lvl,:,:].ravel()[idcs_sort]
        r2 = np.searchsorted(r_grid.ravel(),r[-1])
        urad[lvl],_ = calc_hist(r_sort[:r2],u_sort[:r2], \
            bins=r)
        vtan[lvl],_ = calc_hist(r_sort[:r2],v_sort[:r2], \
            bins=r)
        tlp[lvl],_ = calc_hist(r_sort[:r2],p_sort[:r2], \
            bins=r)
        glp[:],_ = calc_hist(r_sort[:r2],glp_sort[:r2], \
            bins=r)
        sig[lvl],_ = calc_hist(r_sort[:r2],sig_sort[:r2], \
            bins=r)

        if f_id < len(filenames)-1:
            cimf_sort = ds['CIMF'][lvl,:,:]\
                .ravel()[idcsc_sort]
            cimf[lvl],_ = calc_hist(rc_sort[:r2], \
                cimf_sort[:r2],bins=r)
        else:
            cimf[lvl] = np.nan

        ds['latc'] = latc
```

```
        ds['lonc'] = lonc

        np.save(readpath+filenames[f_id],ds)

        avg_data = {
            'lvl':ds['lvl'],
            'tim':ds['tim'],
            'glp':glp,
            'tlu':urad,
            'tlv':vtan,
            'tlp':tlp,
            'sig':sig,
            'cimf':cimf,
            'latc':latc,
            'lonc':lonc,
            'r':(_[1:]+_[:-1])/2
        }

        np.save(writepath+name+'_averaged+%03i.npy'\
            %f_id,avg_data)
```

## B.4 Peak detection

```
import copy

def find_peaks(yin):
    y = np.array(yin)
    ymax = []
    kmax = []
    width = []
    npeaks_max = 5

    # Detect peaks
    while len(ymax) < npeaks_max:
        kc = np.nanargmax(y)
        if kc < 4:
            y[kc] = np.nan
            continue
        if ymax != [] and y[kc] < ymax[0]/2:
            break

        # Determine left end peak
        kl = copy.deepcopy(kc)
        iter_left = True
        while iter_left:  #iterate from kc to left
            kl -= 1
            if kl < 0:
                kl += 1
                break     #stop if k is out of domain
```

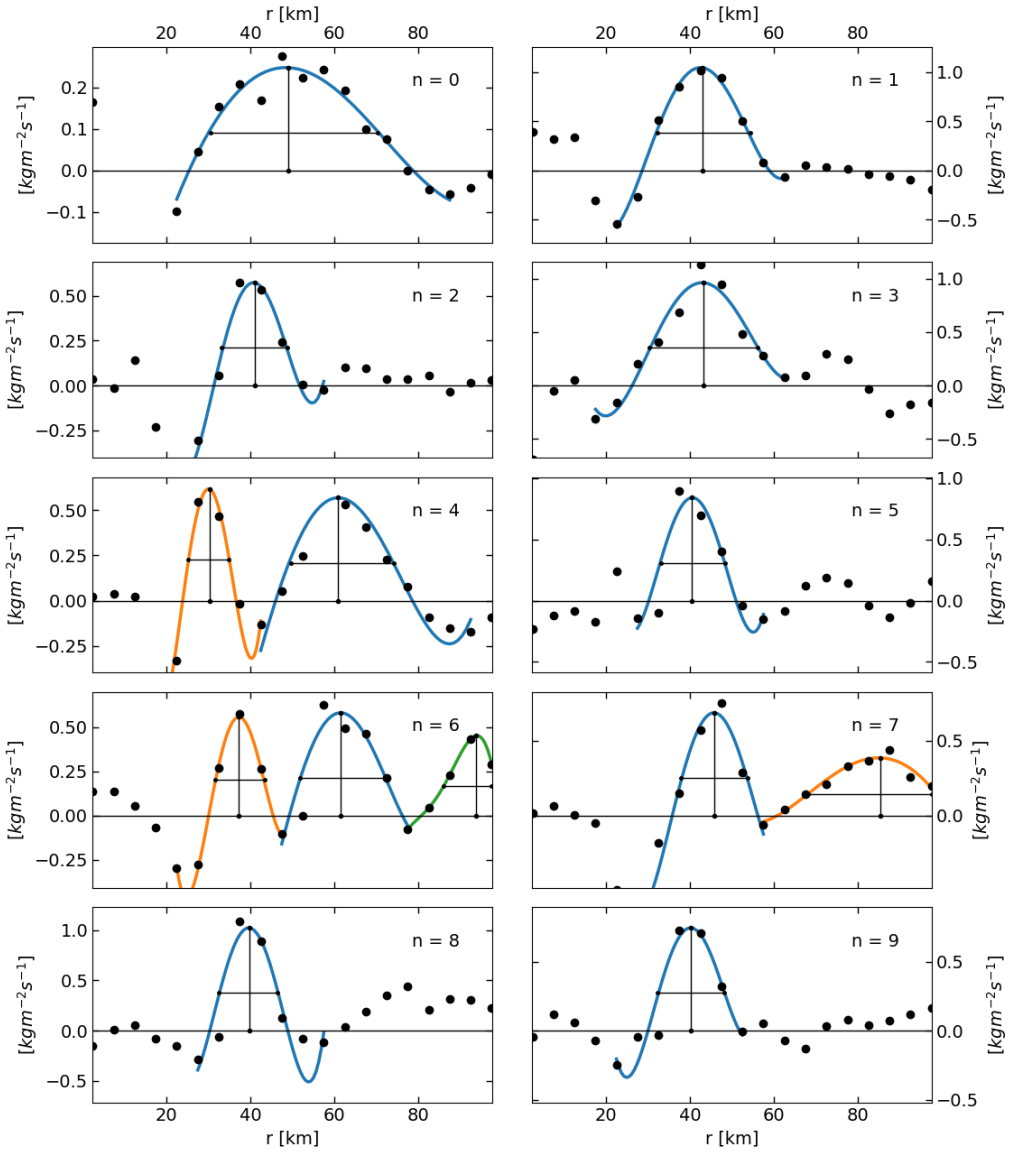


FIGURE B.1: Peak detection

```
        elif y[kl] != y[kl]:
            kl += 1
            break       #stop at nan value
        elif (y[kl] > y[kl+1]) and (y[kl+1] < y[kc]/2):
            kl += 1
            break       #stop if y increases after
                        #'low enough' minimum
        elif kl == 0:
            break       #stop at k=0
        else:
            pass

    #Determine right end peak
    kr = copy.deepcopy(kc)
```

```
        iter_right = True
        while iter_right: #iterate from kc to right
            kr += 1
            if kr > len(y-1):
                kr -= 1
                break     #stop if k is out of domain
            elif y[kr] != y[kr]:
                kr -= 1
                break     #stop at nan value
            elif (y[kr] > y[kr-1]) and (y[kr-1] < y[kc]/2):
                kr -= 1
                break     #stop if y increases after
                          #'low enough' minimum
            elif kr == len(y)-1:
                break     #stop at last k
            else:
                pass

        # Fit peak
        y_peak = np.array(y[kl:kr+1])
        x_peak = np.arange(len(y_peak))
        weights = np.array(y_peak>0) + 1
        fit = np.polynomial.polynomial.Polynomial.fit( \
            x_peak, y_peak, 4, w = weights)
        x_fit = np.linspace(0, kr - kl, 50)
        y_fit = fit(x_fit)

        # Calculate max/width/rmax
        kc_fit = np.argmax(y_fit)
        xc_fit = x_fit[kc_fit]
        ymax_fit = y_fit[kc_fit]
        ylim_fit = ymax_fit/np.exp(1)
        dif = np.abs(y_fit - ylim_fit)
        x1 = x_fit[np.argmin(dif[:kc_fit])]
        x2 = x_fit[np.argmin(dif[kc_fit:])+kc_fit]
        ymax.append(ymax_fit)
        kmax.append(xc_fit+kl)
        width.append(x2-x1)

        # Remove peak from data
        y[kl+1:kr] = np.nan

    return ymax, kmax, width
```

# Bibliography


Bui, H. H. et al. (2009). "Balanced and unbalanced aspects of tropical cyclone intensification". In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 135.644, pp. 1715–1731.

Cangialosi, J. P., A. S. Latto, and R. Berg (June 2018). "National Hurricane Center tropical cyclone report: hurricane Irma". In: URL: https://www.nhc.noaa.gov/data/tcr/AL112017_Irma.pdf.

Charney, J. G. and A. Eliassen (1964). "On the Growth of the Hurricane Depression". In: *Journal of Atmospheric Sciences* 21.1, pp. 68–75.

Delden, A. J. van (1989). "On the deepening and filling of balanced cyclones by diabatic heating". In: *Meteorology and Atmospheric Physics* 41.3, pp. 127–145.

Edouard, S., R. Vautard, and G. Brunet (1997). "On the maintenance of potential vorticity in isentropic coordinates". In: *Quarterly Journal of the Royal Meteorological Society* 123.543, pp. 2069–2094.

Emanuel, K. A. (1986). "An air-sea interaction theory for tropical cyclones. Part I: Steady-state maintenance". In: *Journal of the Atmospheric Sciences* 43.6, pp. 585–605.

— (1997). "Some aspects of hurricane inner-core dynamics and energetics". In: *Journal of the atmospheric sciences* 54.8, pp. 1014–1026.

— (2012). "Self-stratification of tropical cyclone outflow. Part II: Implications for storm intensification". In: *Journal of the atmospheric sciences* 69.3, pp. 988–996.

Emanuel, K. A. and R. Rotunno (2011). "Self-stratification of tropical cyclone outflow. Part I: Implications for storm structure". In: *Journal of the Atmospheric Sciences* 68.10, pp. 2236–2249.

Hendricks, E. A., M. T. Montgomery, and C. A. Davis (2004). "The role of "vortical" hot towers in the formation of tropical cyclone Diana (1984)". In: *Journal of the atmospheric sciences* 61.11, pp. 1209–1232.

Houze Jr, R. A., W-C. Lee, and M.M. Bell (2009). "Convective contribution to the genesis of Hurricane Ophelia (2005)". In: *Monthly Weather Review* 137.9, pp. 2778–2800.

Middleton, J. F. (1987). "Energetics of linear geostrophic adjustment". In: *Journal of physical oceanography* 17.6, pp. 735–740.

Montgomery, M. T., J. Persing, and R. K. Smith (2015). "Putting to rest WISHE-ful misconceptions for tropical cyclone intensification". In: *Journal of Advances in Modeling Earth Systems* 7.1, pp. 92–109.

Montgomery, M. T. and R. K. Smith (2014). "Paradigms for tropical cyclone intensification". In: *Australian Meteorological and Oceanographic Journal* 64.1, pp. 37–66.

Ooyama, K. V. (1964). "A dynamical model for the study of tropical cyclone development". In: *Geofisica Internacional (Mexico)* 4, pp. 187–198.

— (1969). "Numerical simulation of the life cycle of tropical cyclones". In: *Journal of the Atmospheric Sciences* 26.1, pp. 3–40.

— (1982). "Conceptual evolution of the theory and modeling of the tropical cyclone". In: *Journal of the Meteorological Society of Japan. Ser. II* 60.1, pp. 369–380.

Smith, R. K., M. T. Montgomery, and N. Van Sang (2009). "Tropical cyclone spin-up revisited". In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 135.642, pp. 1321–1335.

Van Sang, N., R. K. Smith, and M. T. Montgomery (2008). "Tropical-cyclone intensification and predictability in three dimensions". In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 134.632, pp. 563–582.

Willoughby, H. E. (1990). "Gradient balance in tropical cyclones". In: *Journal of the Atmospheric Sciences* 47.2, pp. 265–274.

Zhang, J. A. et al. (2011). "On the characteristic height scales of the hurricane boundary layer". In: *Monthly Weather Review* 139.8, pp. 2523–2535.