Sequential Data Assimilation and Its Application to Tsunami Analysis in the Japan Sea

Kazuyuki Nakamura

DOCTOR OF PHILOSOPHY

Department of Statistical Science
School of Multidisciplinary Sciences
The Graduate University for Advanced Studies

2006

Abstract

This thesis is organized by two parts. In part I, new data assimilation framework for tsunami simulation models and tide gauge data is introduced. Yamato Rises in the middle of the Japan Sea is analyzed using tide gauge data of Okushiri (Hokkaido–Nansei–Oki Earthquake) Tsunami occurred in the Japan Sea in 1993. The result of this analysis indicates that the Yamato Rises might be shallower than the existing topography data sets. Many physical variables and nonlinear operations of the simulation model cause difficulty in data assimilation. In part II, methods how to manage high dimensionality and nonlinearity are discussed from the viewpoint of data assimilation. In the earlier sections of part II, the management of nonlinearity is discussed through performance comparison among ensemble based filters used in data assimilation. In the later sections of part II, new smoothing scheme is introduced. Numerical experiments are performed using Nikkei 225 stock price data. Management of high dimensionality is discussed to reduce mainly the memory space required from the large-scale data assimilation.

Past studies about tsunami have been based on making and performing numerical simulation models and comparing observed data with simulation results. Bottom topography data set in these simulations is usually fixed as a priori boundary conditions. However, the data sets have errors which cause inaccurate simulation results because propagation speed of tsunami depends on the sea depth. The accuracy of the topography data is insufficient yet regardless of the recent updates attributed to the ship acoustic measurements and satellite altimeter data analysis. Therefore, bottom topography correction is required to simulate tsunami accurately. In addition, bottom topography correction may

improve the knowledges in oceanography and meteorology.

Data assimilation is a technique for a synthesis of information from a dynamic numerical model and observation data especially in geosciences. It is formulated by a state estimation or a parameter identification problem in a state space model. The system and observation models in a state space model correspond to numerical model-based simulations and satellite and/or ground-based measurement systems, respectively. In the new approach, tsunami simulation model and the tide gauge data correspond to the system model and the observation data. Since the observations are sparse in this problem, refinement of parametrization is also introduced.

To validate the method, comparative experiments called identical twin experiments are performed using two kinds of bathymetries; one is artificial bathymetry and the other is the Japan Sea bottom topography. The results demonstrate that the new approach works well for the purpose of correction of inaccurate bottom topography. Based on this framework, the bottom topography is corrected for the Okushiri Tsunami case in the Japan Sea. The effectiveness of the new approach is confirmed and new findings are obtained about Yamato Rises in the middle of the Japan Sea. The estimated topography for the Yamato Rises is shallower than the average of the available data set over most of the area. Additionally, Nihonkai–Chubu Earthquake Tsunami which occurred in the Japan Sea in 1983 is simulated using the original and corrected bottom topography. Tsunami arrival time prediction is improved under the corrected bottom topography.

In the earlier sections of part II, performances are compared between the ensemble filters and smoothers. Numerical experiments show the superiority of the particle filter to the ensemble Kalman filter with the nonlinear observation system. Nextly, the smooth bootstrap technique is discussed in view of relationship between ensemble filters and genetic algorithms. Numerical experiments are also conducted under two types of simulation models. Based on them, applicability of the smooth bootstrap technique in data assimilation is discussed.

In the latter of part II, a new scheme is demonstrated within the particle smoother. Degeneration

problem causes some estimation bias in the particle smoother because resampling from same support is repeated. A straightforward approach to overcome the problem is to increase particles. However, it is difficult because space complexity in the particle smoother is linear order to the time length to be smoothed. The suggested new scheme, called recursive recomputation scheme, can reduce required memory space from linear to logarithmic order maximally. As a result, the number of particles can be increased drastically and length effectively smoothed can be extended. The result of numerical experiment using Nikkei 225 stock index shows desirable aspects of the new scheme in estimation.

Contents

List of Figures

List of Tables

Part I

Data Assimilation for Tsunami Simulation Model

Chapter 1

Introduction

1.1 Motivation

Many studies have been made on tsunamis using various simulation models. These studies have contributed much to our knowledge of geophysics, coastal engineering and disaster prevention. Simulation result of tsunami is essentially dominated by the initial condition and bottom topography. However, the bottom topography data contain errors. For example, the contour plot in Figure 1.1 shows the depth and the ratio of standard deviation to average depth among four sea bottom data sets of the Japan Sea. Used data sets are DBDB-V [NAVOEANO], ETOPO2 [Smith and Sandwell, 1997], JTOPO1 [MIRC, 2003] and SKKU [Choi *et al.*, 2002]. It can be found that the shallow areas, such as the Yamato Rises, which are in the middle of the Japan Sea, and the coastal area, have large differences among the data sets. These differences may cause the errors in estimating the arrival times of tsunamis and other physical quantities, because the propagation speed and direction of a tsunami is largely reflected to the bathymetry. Hence, estimates of the arrival time are more sensitive to differences in shallower zone. Figure 1.2 shows the simulation results of Okushiri (Hokkaido–Nansei–Oki Earthquake) Tsunami based on different bathymetry data set (DBDB-V and SKKU). Not only arrival time of first wave but also the amplitude and the phase are different between other simulations. To obtain better results from simulation models it is important to find a method to correct the errors in

the bottom topography.

In general, a simulation model always has incomplete factors, such as undetermined initial and boundary conditions, unmodeled dynamics, discretization errors and other errors. Incomplete simulation models give inaccurate forecasts. Observation data are often used to clarify measured physical phenomena; in the case of tsunami research, tide gauge records have been intensively used for this purpose. Generally speaking, however, observable variables are limited in spatial and temporal resolution because of cost and technical reasons. Tide gauge records are insufficient because the installation points are not satisfactory and the information gathered by them is limited. Hence the inverse problem cannot be solved using only observations.

Data assimilation (DA, [e.g. Wunsch, 1996; Daley, 1991; Bennett, 2001]) is a concept in geophysics that combines observations with numerical simulation models to compensate for incomplete information in either. Objects of DA are not only to correct the physical variables in the simulation model (interpolation), but also to estimate the parameters, to evaluate model error and sensitivity (modeling). Figure 1.3 shows the summary of various objects in data assimilation. An appropriate evaluation of model error can give a better evaluation of forecast accuracy in geophysical phenomena. Hence, using DA to combine incomplete tsunami simulation model and insufficient tide gauge records will lead to new, useful knowledge about tsunamis and tsunami simulation models.

Several assimilation or inversion methods [Satake, 1989; Tanioka and Satake, 2001; Titov *et al.*, 2005; Tanioka *et al.*, 2006] have been proposed for the study of tsunamis. They concentrate on tsunami source estimation or correction of tsunami height including run-up height near the shore, with the prescribed bottom topography. Tsunami source inversion [Satake, 1989] is a method to correct tsunami height and to clarify tsunami source by solving inverse problem. In this method, simulation model is assigned as transfer function to calculate impulse response and observations of tsunami are assumed to be linear combination of time series of impulse response. The new approach is similar in that the solution is calculated from observed data using a simulation model. However,

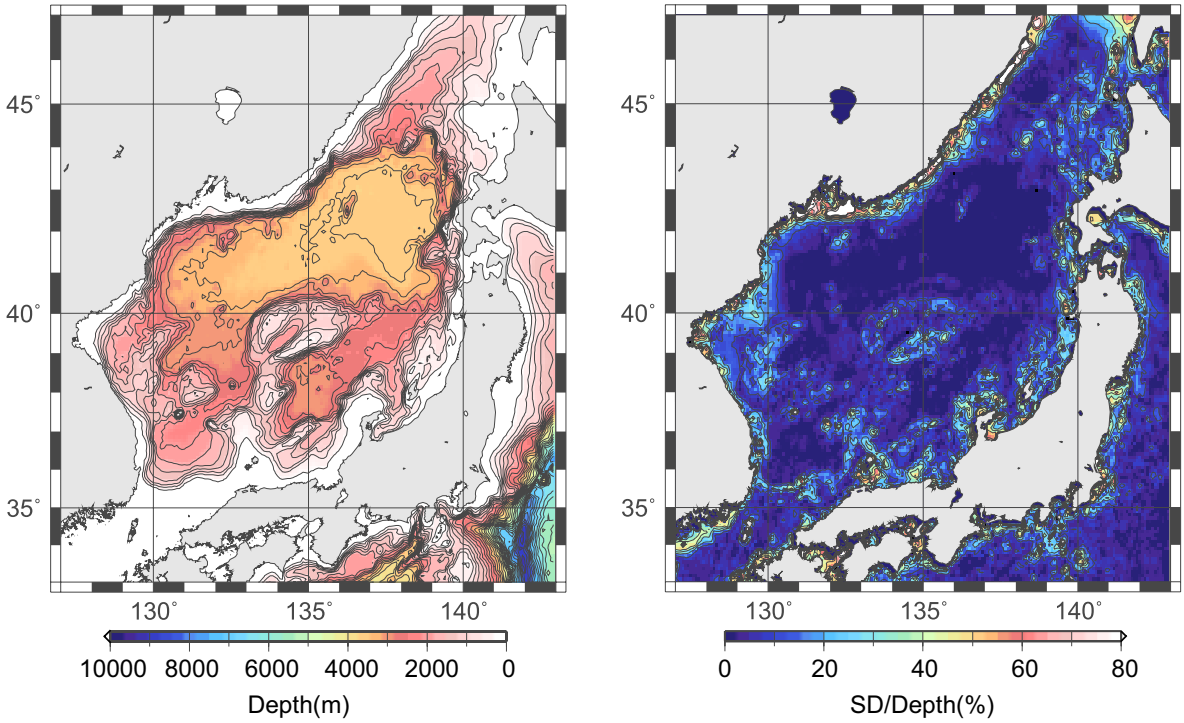

Figure 1.1: Properties of bottom topography data set. The left figure shows the average of the four available bottom topography set. The right figure shows the ratio of the standard deviation among the data sets to depth at each point, represented by the contour map.

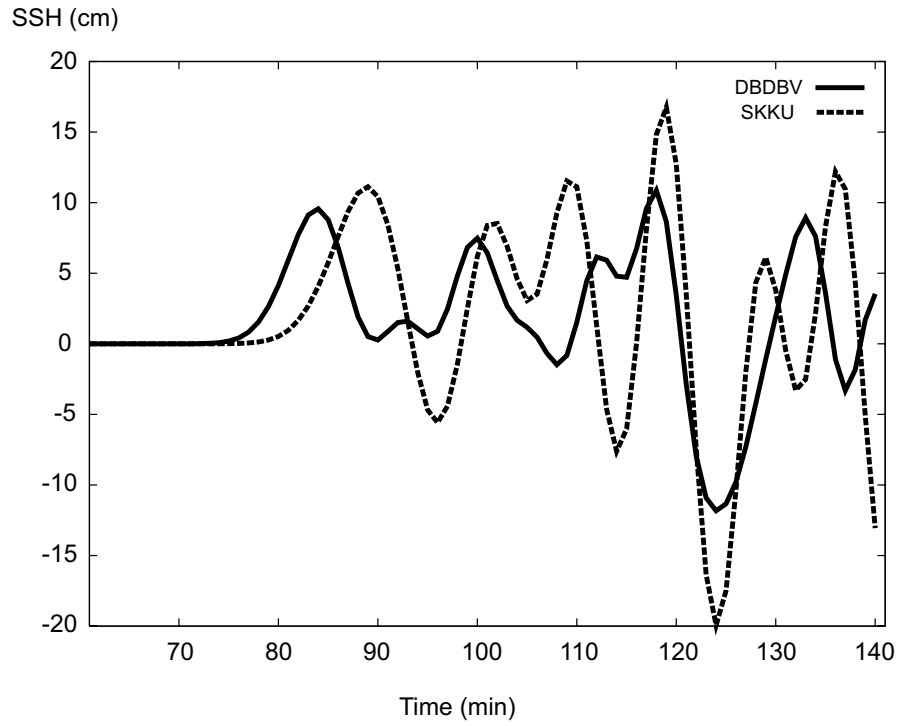

Figure 1.2: Comparison of Okushiri Tsunami simulation under two different bottom topographies.

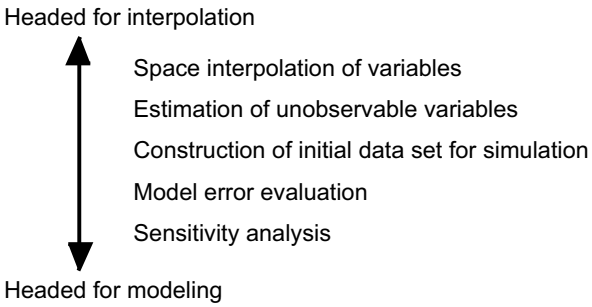

Figure 1.3: Various objects and directions in data assimilation

the new method is different in that it introduces and evaluates errors in the bottom topography as an incomplete boundary condition, which gives more flexible modeling and estimation of physical variables. This approach gives not only effective estimation and prediction but also provides improved new knowledge of the sea bottom useful in geophysics, such as the depths of banks.

1.2 Organization of Part I

The organization of part I is as follows. In Chapter 2, data assimilation are formulated through nonlinear state space model (SSM). Sequential data assimilation is also formulated. In the sequel, the particle filter (PF) which can be applied as a sequential DA procedure is explained. Tsunami simulation model, tide gauge data and the formulation of data assimilation for bottom topography correction are shown in Chapter 3. Introduction of system noise which causes problems in sequential DA are also shown. In Chapter 4, results of numerical identical twin experiments are shown. Two types of experiments, experiments on artificial bottom topography and on the Japan Sea are presented. The results shows the efficiency of the introduced framework. In Chapter 5, sequential DA to correct Yamato Rises in the Japan Sea are applied with use of tide gauge data of Okushiri (Hokkaido–Nansei–Oki Earthquake) Tsunami occurred in the Japan Sea in 1993. Pre-processing procedures of tide gauge records, the result of bottom topography correction and prediction test of Nihonkai–Chubu Earthquake Tsunami are described. Discussion is given in Chapter 6.

Chapter 2

Data Assimilation

As noted in the previous chapter, data assimilation is a concept that combines observations with numerical models. It can be formulated as a state estimation problem by a nonlinear State Space Model (SSM, *e.g.*, [Harvey, 1989; Durbin and Koopman, 2001]). The SSM has given a platform in non-stationary time series and control studies for three decades after Kalman [1960]. In this chapter, a nonlinear SSM is introduced and data assimilation problems are formulated in terms of non-linear SSM.

2.1 Characterization of Data Assimilation

The formulation of a sequential DA problem with an SSM are demonstrated in this section. Additionally, the distinctions between DA and other estimation problems are explained.

2.1.1 Construction of system model

Figure 2.1 shows the procedure of design of DA problems. Partial differential equations (PDE) are usually employed to approximate a real physical system in geophysics. First, the PDE are discretized spatially and temporally for calculation on computer. This discretization gives finite difference equa-

tions (FDE) which can be represented as

$$\boldsymbol{x}_t = \tilde{f}_t(\boldsymbol{x}_{t-1}), \tag{2.1}$$

where $\boldsymbol{x}_t$ denotes all the variables included in the FDE at time step t. Many numerical simulation studies in geophysics and other areas are conducted with solving these equations. Such sets of equations are called simulation models. In the next step, the uncertainties of the simulation model, *i.e.*, the boundary conditions, unknown parameters and some kind of model uncertainties, are modeled by introducing system noise $\boldsymbol{v}_t$. Finally, the simulation model and system noise are combined into an equation

$$\boldsymbol{x}_t = f_t(\boldsymbol{x}_{t-1}, \boldsymbol{v}_t), \tag{2.2}$$

which can be identified with the system model of SSM. f_t is determined from the design how to combine $\tilde{f}_t(\boldsymbol{x}_{t-1})$ and $\boldsymbol{v}_t$. Because the system noise $\boldsymbol{v}_t$ represents "uncertainties", it can be regarded as random variable whose distribution is usually assumed to be normal:

$$\boldsymbol{v}_t \sim N(\boldsymbol{0}, Q_t),$$

where Q_t is pre-determined covariance matrix. In some problems, the initial state $\boldsymbol{x}_0$ also has uncertainties and then can be regarded as random variable.

Compared with usual state estimation problems using SSM, there are several differences in DA problems. One of them is that the simulation model is very large and complicated, often given only by computer source code because we should rely on reservoirs of simulation science in geophysics. Consequently, it is hard to change the simulation model drastically. Another point is the large dimension of the state vector. Each grid point has associated physical variables, which means that the dimension of the state vector is the product of the number of grid points and the dimension of the physical variables. For example, if there are 10 physical variables at each point of a two-dimensional 100×100 grid, the dimension of $\boldsymbol{x}_t$ is 10^5. All the differences make the problems hard. They can

be overcome however by clever design of the uncertainties, including design of system noise and accurate calculation in the prediction steps.

2.1.2 Observation Model

Observations are obtained by the partial measurements of the state vector or their (non-)linear transformations with measurement errors. Therefore, observation equation is written by

$$\boldsymbol{y}_t = h_t(\boldsymbol{x}_t, \boldsymbol{w}_t) \quad \text{(nonlinear observation)}, \tag{2.3}$$

or

$$\boldsymbol{y}_t = H_t \boldsymbol{x}_t + \boldsymbol{w}_t \quad \text{(linear observation)}, \tag{2.4}$$

where y_t is composed of all the measurements, $\boldsymbol{w}_t$ represents measurement error, h_t is nonlinear operator representing observation and H_t represents observation matrix. The PDF of measurement error $\boldsymbol{w}_t$ is usually normal distribution with average $\mathbf{0}$, *i.e.*,

$$\boldsymbol{w}_t \sim N(\mathbf{0}, R_t),$$

where R_t is pre-determined covariance matrix of measurement errors at time t. Equation (2.3) (or (2.4)) is called observation model of SSM. When a measurement is made, a measurement error is included naturally. Therefore, if the transformation and measurement error statistics are known, construction of an observation system is straightforward.

2.1.3 Data Assimilation

The nonlinear SSM is summarized as follows:

$$\boldsymbol{x}_t = f_t(\boldsymbol{x}_{t-1}, \boldsymbol{v}_t),$$

$$\boldsymbol{y}_t = h_t(\boldsymbol{x}_t, \boldsymbol{w}_t),$$

$$\boldsymbol{v}_t \sim N(\mathbf{0}, Q_t), \ \boldsymbol{w}_t \sim N(\mathbf{0}, R_t),$$

where Q_t and R_t are pre-determined or estimated covariance matrices. In the following, the dimension of the state vector $\boldsymbol{x}_t$ is denoted by n_x and the dimension of the observation (measurement) vector $\boldsymbol{y}_t$ is n_y. $\boldsymbol{y}_{1:t}$ is used to represent the set $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_t\}$.

Figure 2.2 shows a schematic representation of the data assimilation concept for the special case, *i.e.*, linear simulation model such that

$$
\begin{aligned}
x_t &= F_t x_{t-1} + v_t, \\
y_t &= x_t + w_t, \\
&\quad (n_x = n_y = 1).
\end{aligned}
$$

At each time the state vector is updated according to this scheme (called the filtering step in the Kalman filter). $\boldsymbol{x}_{t|t-1}$ is a state estimation at time t only via simulations given $\boldsymbol{y}_{1:t-1}$. $\boldsymbol{e}_{t|t-1}$ is a prediction error between an actual observation and predictive value of the observation based on the result of simulations. K_t is a trade off parameter to control how the simulation model accommodates an actual observation. K_t is called the Kalman gain. When $K_t = 0$, an actual observation has no effect on a simulation process. In this case we can totally rely on the simulation result. On the other hand, when $K_t = 1$, any discrepancy between the predictive and real values of observations is perfectly adjusted. In this case, it is difficult to identify a dynamics inherent to a simulation model from an estimation of the state vector, because a state vector is highly sensitive to the observation errors. In general, a filtering procedure estimates $\boldsymbol{x}_t$ from the observed data set $\boldsymbol{y}_{1:t}$ at time t. That is, the procedure estimates the most likely value of the vector $\boldsymbol{x}_t$, or the PDF, as the time series data $\boldsymbol{y}_t$ are observed. If both (2.2) and (2.3) are linear equations, the KF is efficient and accurate. However, it is almost impossible to estimate $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ accurately if one of the equations or the noise term is nonlinear.

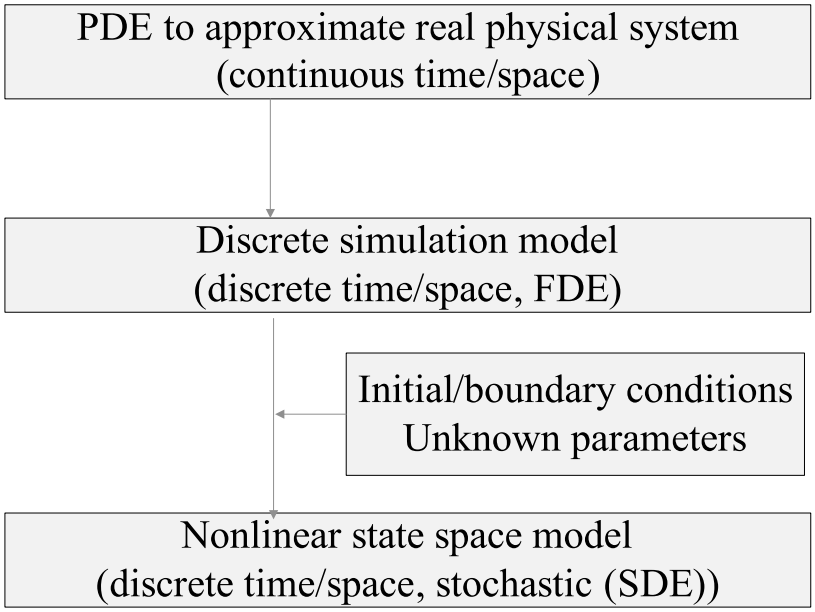

Figure 2.1: Procedure for constructing system model.

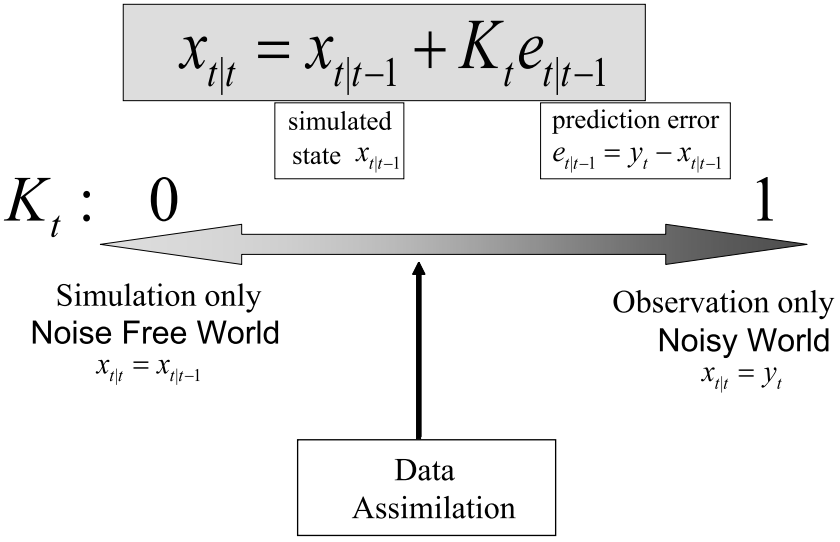

Figure 2.2: Schematic representation of data assimilation.

2.2 Particle Filter in Sequential Data Assimilation

An approach to overcome the difficulty in a nonlinear SSM estimation is Monte Carlo approximation such as the particle filter (PF, [Kitagawa, 1996; Gordon *et al.*, 1993]). If the DA problem can be formulated in the context of an SSM, the particle filter is applicable. In the PF, the estimated mean and variance which appear in the Kalman filter are replaced by obtaining the set of realizations sampled from PDFs. This realization set is called ensemble and each realization is called ensemble member in the context of the EnKF. In the context of the PF, each realization is called particle. In the following, these representation are used. Ensemble represents approximation of PDFs:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}) \cong \frac{1}{N}\sum_{i=1}^{N}\delta(\boldsymbol{x}_t - \boldsymbol{x}_{t|t-1}^{(i)}),$$

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \cong \frac{1}{N}\sum_{i=1}^{N}\delta(\boldsymbol{x}_t - \boldsymbol{x}_{t|t}^{(i)}),$$

where $\{\boldsymbol{x}_{t|\cdot}^{(i)}\}_{i=1}^{N}$ is ensemble of $\boldsymbol{x}_{t|\cdot}^{(i)}$, $\boldsymbol{y}_{1:t} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_t\}$ and N is the number of particles (ensemble members). $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})$ is called predictive PDF, $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ is called filtered PDF and corresponding sets of ensembles are called predictive and filtered ensemble respectively. The filtering problem is how to estimate and update the ensemble at each time step t using $\boldsymbol{y}_t$. The filtering procedure consists of two step, the prediction step and the filtering step. These two steps are calculated in turn.

In the prediction step, $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^{N}$ is obtained from $\{\boldsymbol{x}_{t-1|t-1}^{(i)}\}_{i=1}^{N}$ and system model (2.2). The ensemble $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^{N}$ is given by the following Monte Carlo simulation:

$$\boldsymbol{x}_{t|t-1}^{(i)} = f_t(\boldsymbol{x}_{t-1|t-1}^{(i)}, \boldsymbol{v}_t^{(i)}), \quad \boldsymbol{v}_t^{(i)} \sim N(\boldsymbol{0}, Q_t).$$

In the DA, $\boldsymbol{v}_t^{(i)}$ corresponds to undetermined boundary conditions and unmodeled dynamics of simulation model above.

In the filtering step $\{\boldsymbol{x}_{t|t}^{(i)}\}_{i=1}^{N}$ is obtained from $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^{N}$ and $\boldsymbol{y}_t$. At first, weight $q_t^{(i)}$ of each ensemble member is calculated from the observation system and the observation $\boldsymbol{y}_t$:

$$q_t^{(i)} = p(\boldsymbol{y}_t|\boldsymbol{x}_t = \boldsymbol{x}_{t|t-1}^{(i)}). \tag{2.5}$$

If the observation system is linear form (2.4), calculated weight is the following form:

$$q_t^{(i)} = |(2\pi)^{n_y} R_t|^{-\frac{1}{2}} \exp\left(-2(\boldsymbol{y}_t - H_t \boldsymbol{x}_{t|t-1}^{(i)})^{\mathrm{T}} R_t^{-1} (\boldsymbol{y}_t - H_t \boldsymbol{x}_{t|t-1}^{(i)})\right).$$

After this calculation, filtered ensemble $\{\boldsymbol{x}_{t|t}^{(i)}\}_{i=1}^{N}$ is made by sampling with replacement from predictive ensemble $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^{N}$ in proportion to the weight $q_t^{(i)}$.

Chapter 3

Data Assimilation for Tsunami Simulation Model

In this chapter, data assimilation framework for tsunami simulation model is introduced. At first, simulation model and observation data used are explained. In next, the way how to introduce uncertainties in the simulation model is explained.

3.1 Simulation Model and Observations

The tsunami simulation model is based on the shallow-water equations [*e.g.*, Choi and Hong, 2001]:

$$\frac{\partial}{\partial t}(Hu) + \frac{\partial}{\partial x}(Hu^2) + \frac{\partial}{\partial y}(Huv) - fHv + gH\frac{\partial \eta}{\partial x} = -\tau u,$$
$$\frac{\partial}{\partial t}(Hv) + \frac{\partial}{\partial x}(Huv) + \frac{\partial}{\partial y}(Hv^2) + fHu + gH\frac{\partial \eta}{\partial y} = -\tau v,$$
$$\frac{\partial \eta}{\partial t} + \frac{\partial}{\partial x}(Hu) + \frac{\partial}{\partial y}(Hy) = 0,$$

where u and v are components of flow vector, f is Coriolis parameter, g is a gravitation acceleration, η is an elevation of sea surface, H is sum of the η and depth d, and tan is the bottom frictional parameter. These continuous equations are discretized spatially and temporally by the finite difference scheme, with use of 5 minute staggered (leap-frog) C grid [Arakawa and Lamb, 1977]. Discretization produces a two-dimensional lattice and each point m of the lattice has four physical scalar variables.

The variables are depth d_m, sea surface height η_m, which is measured from the average sea surface, and the two components of the two-dimensional horizontal velocity components (u_m, v_m) (*see*, Fig. 3.1). The set of all depth variables d_m represents the bottom topography. There are two types of boundary conditions in the simulation model. Non-reflecting boundary conditions are imposed on the edge points of the lattice. The boundary conditions at the border between land and sea are set to no-slip, no-flow. The initial conditions are the initial values of d_m, η_m, u_m and v_m. The initial values of η_m are determined from information on land slip, which is estimated from observed data on an earthquake. The initial values of u_m and v_m are set to zero. The depth d_m of each grid point m is taken from the bottom topography data set; however, this data set is known to be erroneous and is to be re-determined.

Though bottom friction parameter τ could affect propagation of tsunami, it does not occur in the following experiments. To confirm this, Okushiri Tsunami is simulated under the condition that τ is changed from 10 to 1.0×10^{-4}. In this setting, arrival times are not changed in time scale of a minute. The reason is that time scale of analyzed tsunamis (several hours) is short and tsunami propagation, especially arrival time, is not affected by τ. Therefore, the friction parameter can not be assimilated in this study.

The data set used are derived from tide gauge records. Each tide gauge station records a one-dimensional time series of SSH near the installation point. The installation point of tide gauge are shown in Figure 3.2.

The observation vector $\boldsymbol{y}_t$ consists of measurements at each time step. Hence each component of $\boldsymbol{y}_t$ corresponds to the tide gauge time series of each station. Measurement errors are determined from the observed series.

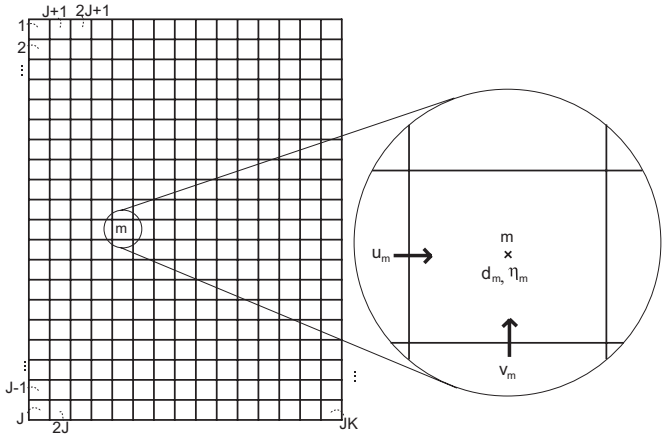

Figure 3.1: Grid of simulation model and physical variables.

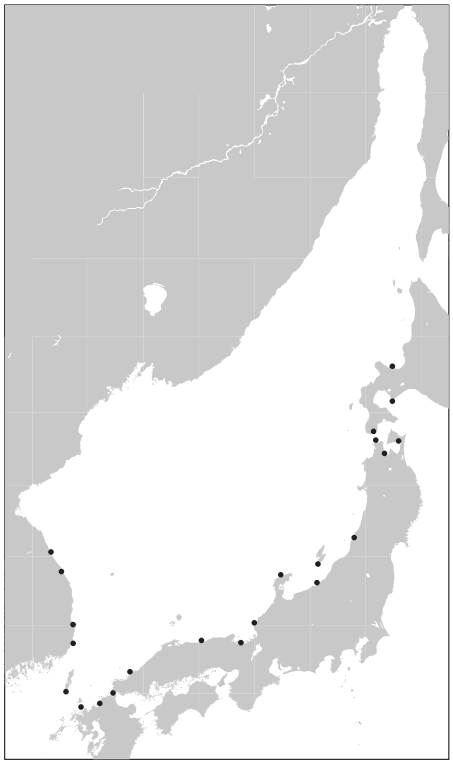

Figure 3.2: Installation point of tide gauge

3.2 Introduction of Uncertainties

Introduction of distribution of initial state vector and system noise is required in estimation of bottom topography. I regard only erroneous part of sea depth d_m as random variable and deal with other variables as deterministic one. It corresponds to introducing probability distribution functions into the components of initial state vector $\boldsymbol{x}_0$ corresponding to erroneous part. Pre-determined values are used for the other components such as the components of SSH or flow vectors especially around the tsunami source. As for system noise, introduction of system noise into the components of depths and/or other physical variables is inappropriate in tsunami case because of the following two reasons; one is that the bottom topography should be regarded as time invariant parameter, the other is that the number of other physical variables is too large compared with the number of observation points to introduce system noise. Therefore, the system noise is not introduced in principle.

In this framework, the time invariant parameters can be determined or the model which is suitable for observations can be identified [Nakamura *et al.*, 2005]. Belief about bottom topography gradually changes and solidifies as observations of tsunami increase. From the viewpoint of statistical estimation problem, prior distribution about bottom topography is introduced at time step 0 and posterior distribution is calculated using all the observed time series $\boldsymbol{y}_{1:T}$. It seems well, however, applying the PF to such a system model can give rise to degeneracy, which causes estimation bias. Therefore, some components in state vector could be disturbed using some "appropriate" system noise. It is referred to self-organizing model [Kitagawa, 1998]. It should be noted that this noise is not derived from modeling, but technical reason. Therefore, choice of system noise should be taken care of because it may give irrelevant freedom. System noise was introduced in Okushiri Tsunami analysis, whereas it did not in other experiments. I explain details in the corresponding subsection.

3.3 Summary

Figure 3.3 shows the procedure how to update the bottom topography in sequential data assimilation in one time step. Simulation step makes candidates of sea surface using shallow-water equations model. If the system noise is added into the sea depth $d_.$, sea bottom also changes. In the filtering step, all the variables including sea depth $d_.$ and sea surface $\eta_.$ change.

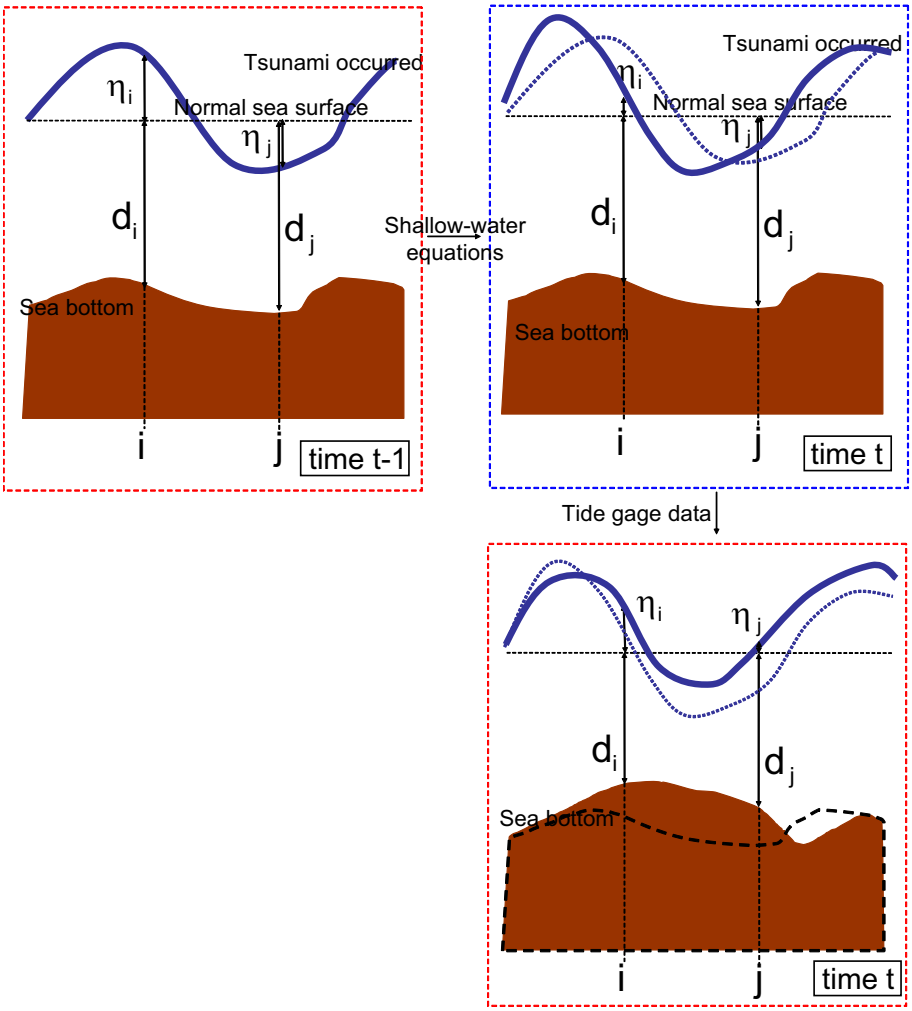

Figure 3.3: Procedure of time step t

Chapter 4

Results of Numerical Experiment

4.1 Identical Twin Experiment

To validate the correction ability of the assimilation methods, numerical experiments using test data are conducted. Figure 4.1 shows procedure of identical twin experiments. It is also referred as "perfect model" experiments. At first, a simulation is run and the results of the simulation are recorded. This is called the model-run stage in the following. Next, an observation data set is constructed by an observation model and simulation results. These simulation results are considered "true", that is, the observation data obtained here are regarded as coming from the "true" model or physics. Consequently, state vectors and parameters are estimated from the observation data set by sequential DA, referred to as the assimilation stage. Finally, we check whether the estimations approach to the "true" simulation results. In the following, I experimented two types of identical twin experiments in which settings about bottom topography are changed; on artificial bottom topography and on the Japan Sea.

4.2 Experiment on Artificial Topography

In this subsection, I show the result of identical twin experiment using artificial bottom topography. Left image in each figure in Figure 4.2 shows the "true" topography in simulation area with propa-

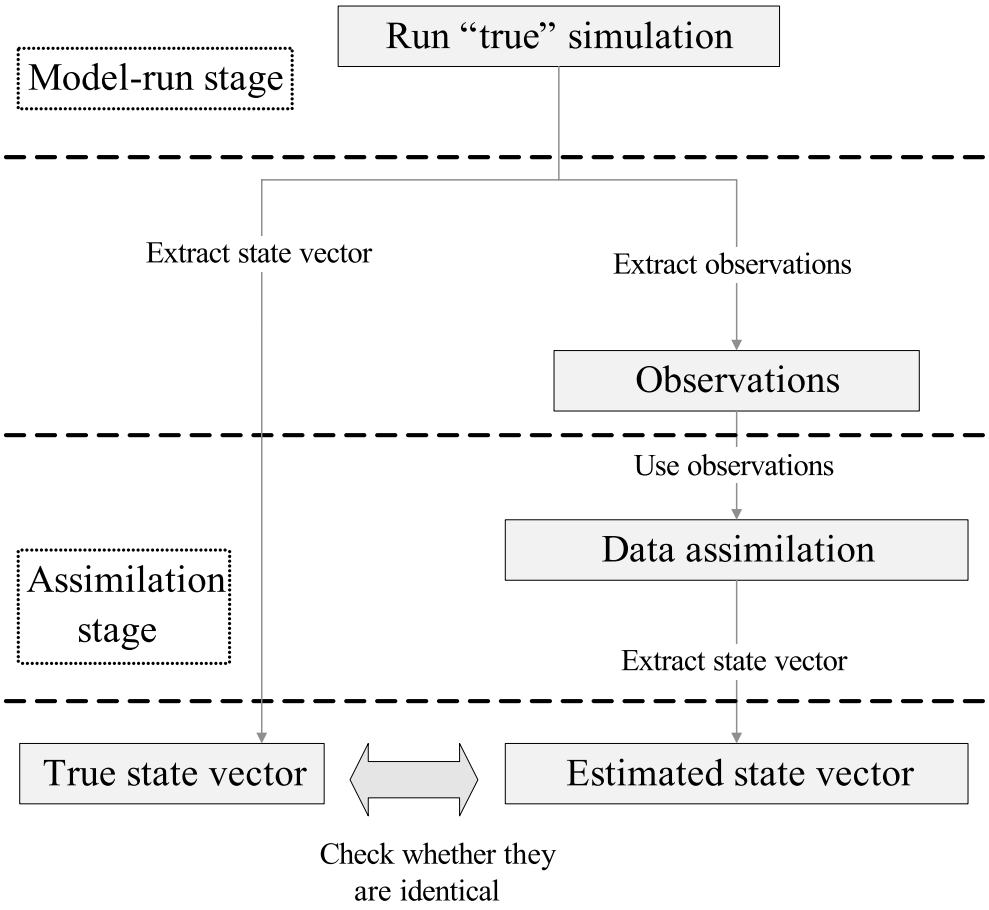

Figure 4.1: Procedure of identical twin experiment.

gation of tsunami. Four observation points are set in left coast represented by white dots in Figure 4.2. Observation model is linear model (2.4):

$$\boldsymbol{y}_t = H_t \boldsymbol{x}_t + \boldsymbol{w}_t,$$

where H_t is a $4 \times n_x$ zero-one matrix and $\boldsymbol{w}_t \equiv 0$. Therefore, the dimension of the observation vector is four. Eight rises can be seen and no rise is in the middle of the simulation area.

At first, a tsunami is simulated. A source is set in the right middle of the area and the set of observation series is made by the SSH at each observation points. In next, another "false" bottom topography which has rise in the middle is determined and initial condition about bottom topography is disturbed. More precisely, d_m is approximated using the sample set $\{d_m^{(i)}\}_{i=1}^N$, which is generated by

$$d_m^{(i)} = \begin{cases} max(\hat{d}_m - c^{(i)}\delta_m, 50) & \text{(false rise area)} \\ d_m & \text{(other area)} \end{cases},$$

where $c^{(i)} \sim N(1.0, 1.5^2)$, $\hat{d}_m$ is the "true" depth at m and δ_m is height of rise at each grid point m. The goal of this experiment is whether the false bank is identified as false one, which means that the middle area is flat.

The result presented in Figure 4.2 shows that the estimation works well. Right chart in each figure represents the profile along white line in left image. Red curve represents sea surface height, green curve represents the true bottom topography, blue curve is estimation, orange and purple curves are shallowest and deepest samples from the ensemble. As tsunami arrives at the observation points in the left coast, the center bottom topography starts to approach to true one. The shallowest and deepest sample are also converging to true one.

4.3 Experiment on the Japan Sea

I used the Okushiri Tsunami for another numerical experiment. This area is discretized longitudinally and latitudinally in a 192(longitude) $\times$ 240(latitude) grid. About half of the grid points are in the sea

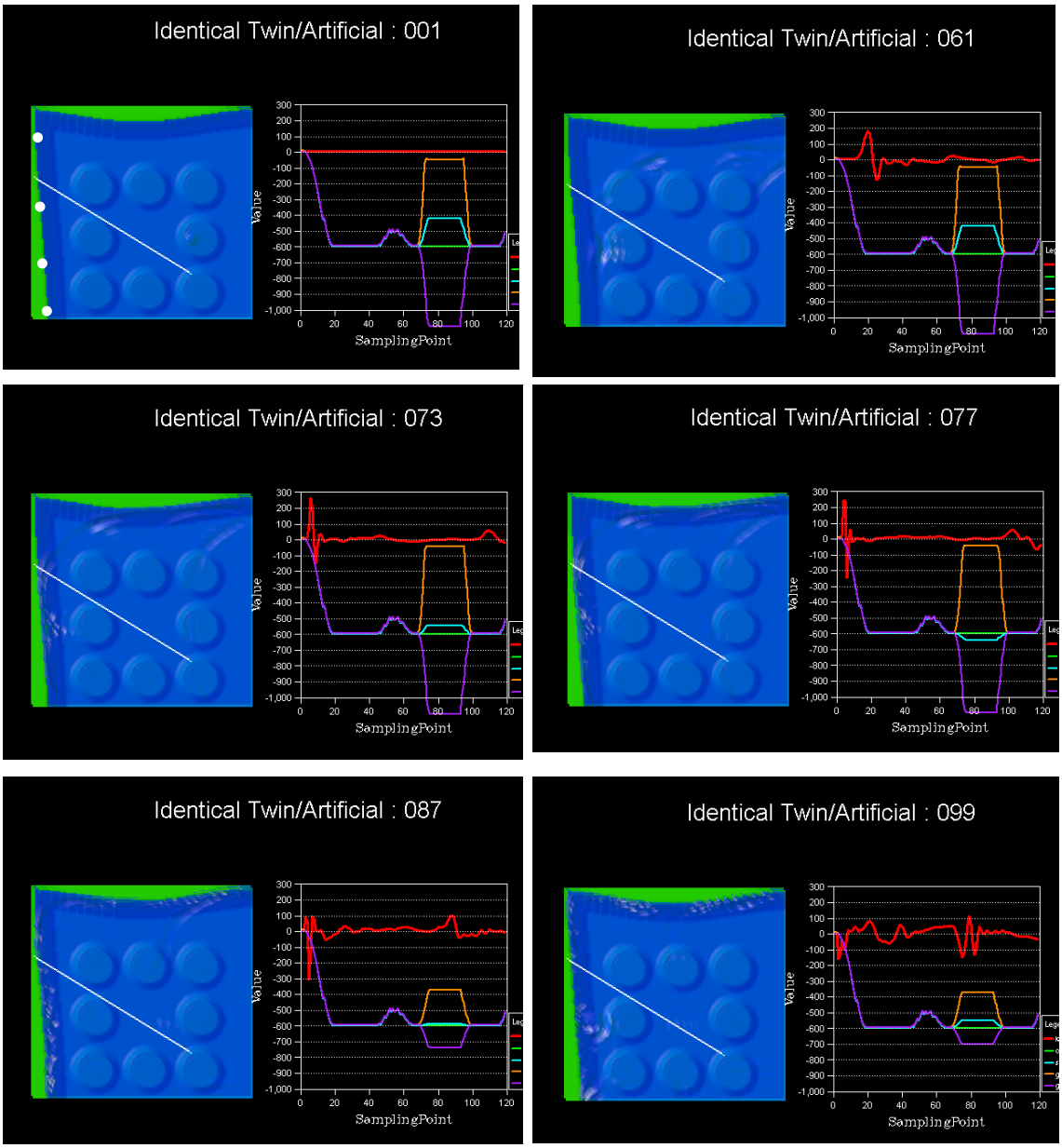

Figure 4.2: Result of identical twin experiment in artificial bottom topography.

and the number of the state of each grid point on the sea is four; therefore, the dimension of the state vector is about 9×10^4. The dimension of the observation vector is also four. The four observation points are illustrated in Figure 4.3.

In the model-run stage, I fixed the bottom topography using a data set and ran the Okushiri Tsunami simulation. The observation data were then generated using the set of SSH time series at each observation point through the simulation. This is equivalent to determining the observation model from the linear model (2.4):

$$\boldsymbol{y}_t = H_t \boldsymbol{x}_t + \boldsymbol{w}_t,$$

where H_t is a $4 \times n_x$ zero-one matrix and $\boldsymbol{w}_t \equiv 0$.

As noted earlier, the bottom topography is time invariant. Therefore, allowing only distribution of the initial conditions is more natural for its parametrization. I distributed each d_m part of $\boldsymbol{x}_0$ and set $\boldsymbol{v}_t \equiv \boldsymbol{0}$ in the assimilation stage. To check the effectiveness of the error correction, I set an initial bottom topography estimation biased from that of the model-run. More precisely, d_m was approximated using the sample set $\{d_m^{(i)}\}_{i=1}^N$. This sample set was generated by

$$d_m^{(i)} = c^{(i)} \hat{d}_m,$$

where $c^{(i)} \sim N(1.1, 1.5^2)$ and $\hat{d}_m$ is the "true" depth at m. This setting means that the initial estimation depth is deeper than the "true" depth and the degree of bias from the "true" one is uniform regardless of grid point. This is equivalent to generating topography samples whose average is biased from the "true" values and whose distribution is modified at every filtering step. The number of particles is set to 100.

Fig. 4.4 shows the result of bottom topography correction. The left side of each image shows the state of the tsunami and the right chart in each image shows the estimated bottom topography at the corresponding time step. The number above each image shows the number of six-minute intervals passed. The curves drawn in the graphs on the right side of each image show the sea surface height

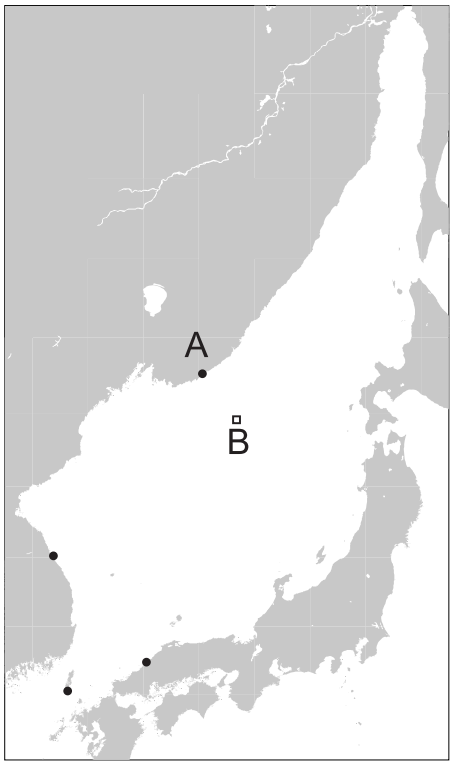

Figure 4.3: Area of identical twin experiment. The installation points of the tide gauge are shown by dots. The check point for water depth is shown by the square marked B

(red curve), the bottom topography of the shallowest sample (orange), the “true” bottom topography (green), the estimated bottom topography (blue) and the bottom topography of the deepest sample (purple) along the white line of the left side of the image.

The results of an observed SSH at an observation point on the Russian coast (point A in Figure 4.3) are shown in Figure 4.5. The estimated and true water depths through all time step in the middle of the Japan Sea (point B in Figure 4.3) are also shown. Point A is the first arrival point of the tsunami amongst the observation points. Immediately after the arrival of the tsunami at point A, the estimated water depth starts to converge to the true water depth.

The time evolution of the deepest and shallowest values of $d_m^{(i)}$ at point B are plotted in Figure 4.6. The range between the largest and the smallest values shrinks and converges to true water depth, indicating that the reliability of the bottom topography estimation improves over time. Therefore, the result of the identical twin experiment demonstrates that the bottom topography can be effectively corrected by using tide gauge data under the condition of real tsunami and bottom topography.

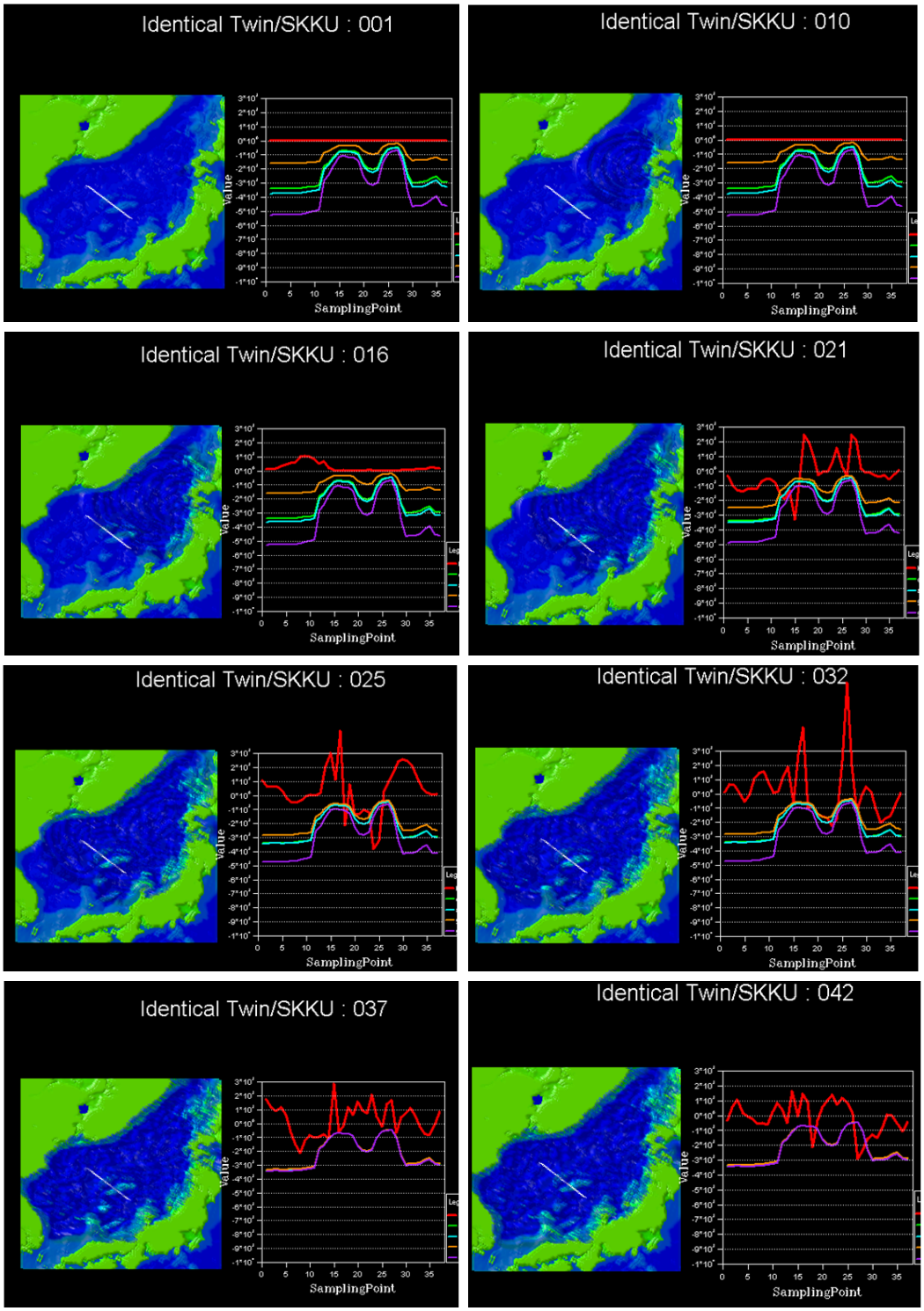

Figure 4.4: Result of identical twin experiment

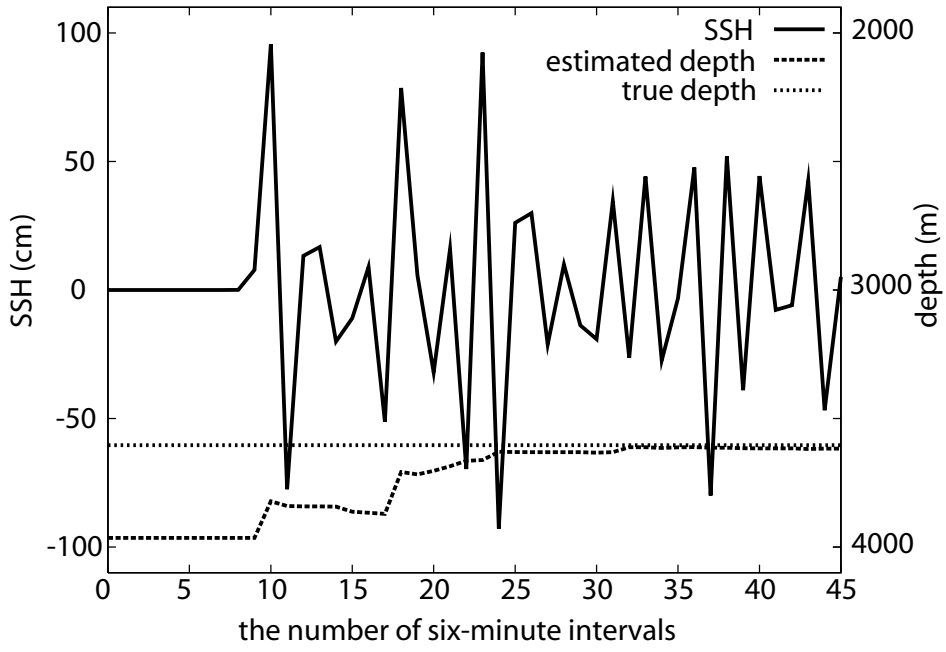

Figure 4.5: Time series of observed SSH at point A and estimation of water depth at point B in Figure 4.3.

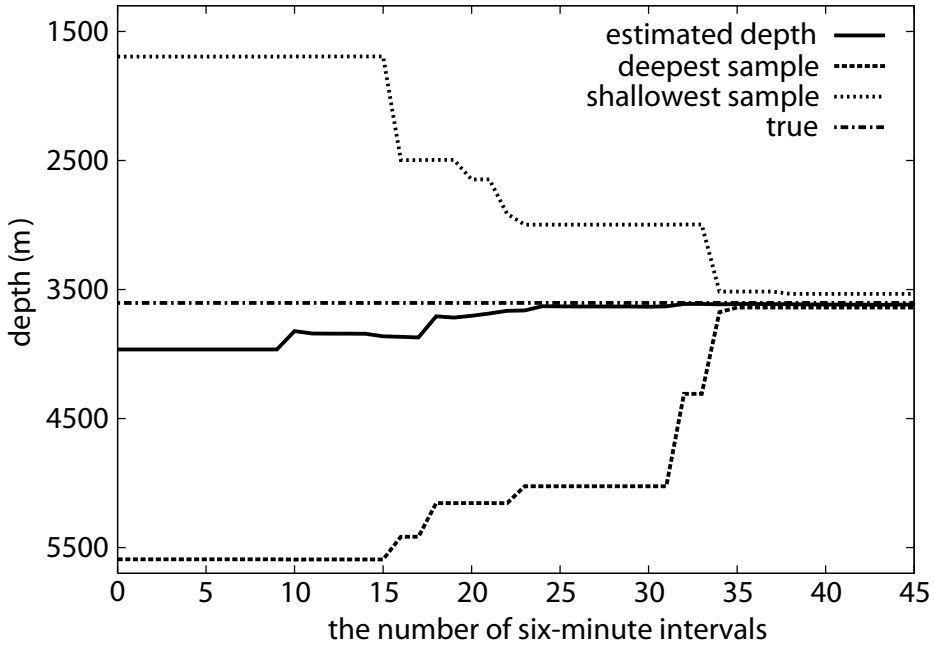

Figure 4.6: Estimation of water depth at point B in Figure 4.3. The deepest and shallowest samples of water depths are also shown.

Chapter 5

Results of Data Assimilation

Based on the method introduced in the previous chapters, bottom topography of the Japan Sea is corrected. Used tide gauge data are records of Okushiri Tsunami (Hokkaido–Nansei–Oki Earthquake Tsunami). These data are pre-processed to apply the method. In section 5.1, pre-processing method is discussed. In next, bottom topography around Yamato Rises are analyzed. The setting and result are shown in section 5.2. In section 5.3, Nihonkai–Chubu Earthquake Tsunami is simulated based on the average and modified topographies.

5.1 Preprocessing Method for Tide Gauge Data

To construct an observation vector, the following procedure is applied. The tide gauge stations are numbered and each tide gauge record is decomposed into two series, trend and residual series. Then the value of the residual series of tide gauge station j at time t is allocated to the jth component of the observation vector y_t. Hence the dimension of the observation vector is the number of tide gauge stations. Two decomposition methods are used, a state space model (SSM) based method and a spectral method. An SSM based method involves fitting to a one dimensional random walk model [Kitagawa and Gersch, 1996; Durbin and Koopman, 2001]. A spectral method is Singular Spectrum Analysis (SSA, Broomhead and King [1986]; Vautard and Ghil [1989]), which is used

to decompose a time series such as ENSO time series. Figure 5.1 shows a trend estimation of tide gauge data of Okushiri Tsunami at Asamushi. Tide gauge record is mainly made of astronomical tide, meteorological tide, harbor oscillation, other trend component and noise. We can observe the small pseudo-periodic component with several or several tens of minutes (harbor oscillation) and the large astronomical tide component in Figure 5.1.

By the methods I have presented, we can remove the astronomical tide, the meteorological tide and the trend component. Other components are left because others except for tsunami component can be managed as observation noise in observation model. The astronomical tide is mainly combination of components which have period of a half day or a day approximately. Periods of these components are longer than the tsunami and the harbor oscillation. Therefore, the astronomical tide is easily removed. The meteorological tide and other trend components also change slowly compared with the components left. Therefore, if hyper-parameters are determined "appropriately," both SSA and an SSM based method can be applicable.

In SSA, used window length is 30 and the largest components used and removed are two. The choice of components is not hard from the viewpoint of removal of rough trend because SSA is a spectral method using PCA and as a consequence, methods and tools to discriminant trend components from others, especially oscillations, are naturally introduced. For example, if tsunami and noise components are small compared with trend component, we can remove a rough trend component estimated by two or three largest singular components.

On the other hand, an SSM based method gives more flexible trend because the fitted model is one dimensional random walk model. We can determine hyper-parameters by Kalman filter under maximum likelihood condition. However, it is inappropriate for this purpose because the model is too flexible to obtain rough trend. $\sigma^2 = 0.005$ is good to detide in this series though the maximum likelihood estimation of σ^2 is larger. Figure 5.2 depicts residual series obtained by subtraction of trend component from tide gauge time series with $\sigma^2 = 0.005$.

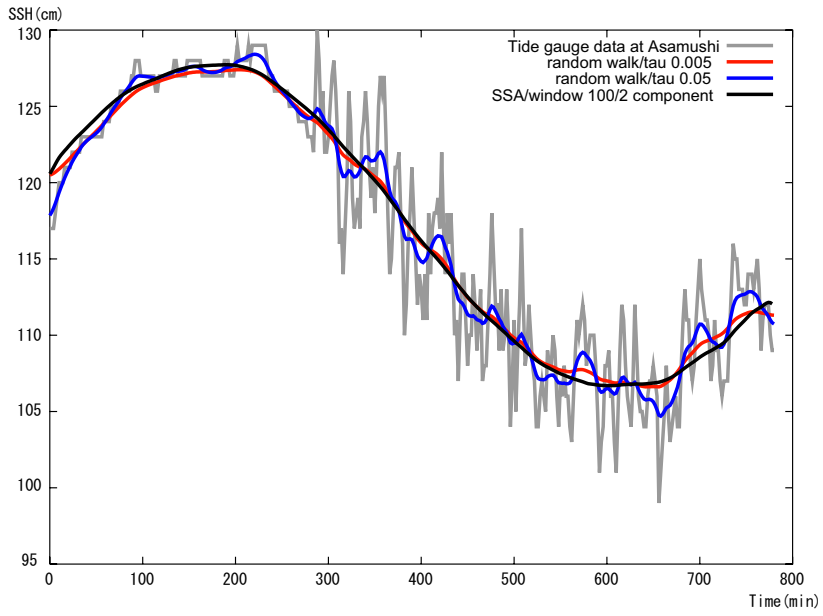

Figure 5.1: Example of tide gauge data and estimation of trend component.

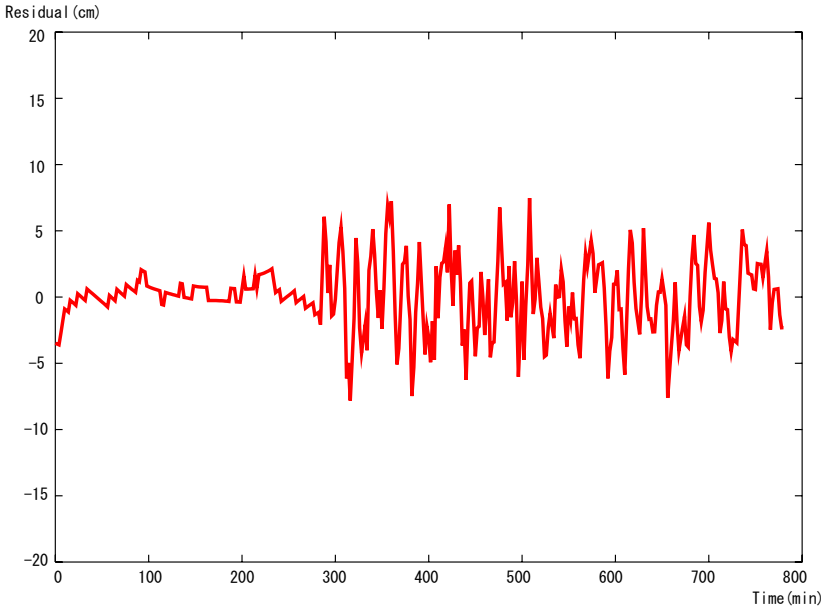

Figure 5.2: Example of tide gauge data and estimation of trend component.

Both methods give similar results, so either can be used. However, different properties exist between SSA and the SSM based method. SSM based method with small variance can remove trend, it is sufficient so far. In the following, the SSM based method is employed for detide. However, SSA can remove periodic or smooth trend, and if we know that some periodic component is not tsunami oriented, we can remove it. Because of these properties, spectral method may be more appropriate in normal situations. However, if there is an aperiodic component such as a height change affected by a change of air pressure, it is not easy to remove by SSA whereas SSM based method can remove this component. Therefore, it depends on data and situations which method is appropriate.

5.2 Bottom Topography Correction

I conducted two types of numerical experiments. The first experiment concerned bottom topography correction and the second concerned calculating the predictive distribution of the arrival time of a tsunami. The Okushiri Tsunami was used to correct the bottom topography. The settings of the simulation model are the same used in the identical twin experiment in the Japan Sea.

Bottom topography data sets have been made from many observation sources. DBDB-V was based on DBDB5 which is 5 minutes bathymetric data developed by the US Naval Oceanographic Office from echo sounder maps in 1980s. DBDB-V is modified version of DBDB5 and have finer resolution in some coastal regions. ETOPO2 is developed from DBDB-V using the correlation between bottom topography and marine gravity anomaly which can be obtained by satellite altimetry data. JTOPO1 is also based on the ship and satellite altimetry data around Japan though their quality have been controlled originally. SKKU is constructed by SungKyunKwan University which aims at obtaining precise bathymetry around Korean Peninsula.

The assimilated area and observation points are illustrated in Figure 5.3. The number of tide gauge stations was 18, and hence the dimension of the observation vector n_y was 18. The decomposition method of each observation time series was to fit a one dimensional random walk model.

When we eliminate trend components from tide gauge data, weather in that time should be taken care of because it affects estimation of trend component. From Japan Meteorological Agency [1993] data, weather around the tide gauge installation points was almost cloudy. The record of wind speed was almost under three meters per second and up to five meters per second around installation points. They did not change drastically, therefore, extra preprocess is not necessary. The observation model was a linear equation (2.4), that is, H_t was a $18 \times n_x$ matrix. The elements of H_t were determined by comparing the maximum height of the tsunami simulation and observations. It is assumed that observation noise among different tide stations are uncorrelated and *iid* in time. The initial conditions of the SSH were also determined from data on the Hokkaido–Nansei–Oki Earthquake.

Because the south rise area of the Yamato Rises have large errors, I set the area and surroundings as the uncertain area and introduce uncertainties as stated earlier. Area A, indicated by the thick line in Figure 5.3, is the uncertain area. I made a set of randomly weighted linear combinations of these topographies and set the combination to be an initial bottom topography sample such that the ith ensemble of the estimated depth $\hat{d}$ at point m and time 0 was determined by

$$\hat{d}^{(i)}_{m,0} = \begin{cases} \sum_{l=1}^{4} w_l^{(i)} d_m^l & \text{(uncertain area)} \\ \sum_{l=1}^{4} 0.25 d_m^l & \text{(other)} \end{cases}, \tag{5.1}$$

where $w_l^{(i)}$ was the ith weight ensemble of the lth topography and was distributed as $w_l^{(i)} \sim N(0.25, \sigma^2)$. It is important to note that $w_l^{(i)}$ does not depend on m, the sub-fix of d, in this formulation. This means that the depth was determined only from estimated coefficients. As a result, the degree of freedom is sufficiently reduced. The number of ensembles is 100. Figure 5.4 shows snapshots of the profile along the segment FG in Figure 5.3. Each panel show the estimated bottom topography at the initial ($t = 0$), medium ($t = 150$) and last ($t = 510$) time steps of the assimilation. The green line represents the estimated topography and the pink line represents the average from the available topographies. The red and blue lines represent the deepest and shallowest candidates of the bottom topography which is represented by the maximum and minimum particle. Roughly speaking, the shallowest value at a point corresponds to a minus deviation and the deepest value corresponds to a

plus deviation.

Figure 5.5 shows contour plots of the difference between average and estimated bottom topography. The left panel shows the difference between the average bottom topography and the estimation around the Yamato Rises at each point. The next two are the subtractions of the shallowest and the deepest candidate from the average at each point. Figure 5.6 shows the time series of the result of the bottom topography estimation at the points 40°N and 135°E, and 39°N and 135°E.

From these results, we can find the following three points. The first point is that major correction of the bottom topography starts about 120 minutes after the occurrence of the tsunami. I estimate that this time is the first arrival time of the wave passing through some part of the disturbed area, because the arrival time of the first wave at Wajima (point E in Figure 5.3) is 90 minutes after occurrence. In addition, minor correction starts about 70 minutes after the occurrence. The reason is that if true bottom topography was deeper than the average, the wave would arrive at observation points faster. However, it does not arrive and therefore deeper estimations are rejected. These points support that the method works properly.

The second point is that the estimated bottom topography is shallower than the average of the four bottom topographies in almost all areas. In addition, the shallowest and deepest candidates of the bottom topography converge, though variances still remain. This means that the error of the estimation has reduced and that the estimation works well in spite of the large dimensionality of the simulation model. The third point is that the south slope of the South Rise is estimated to be deeper than the average of the topographies. Not only the estimation in this area, but also the shallowest particle which corresponds to error band value is deeper than the average. It suggests this aria is deeper than the average with high probability. The second and third points should be investigated further, through more extensive measurements.

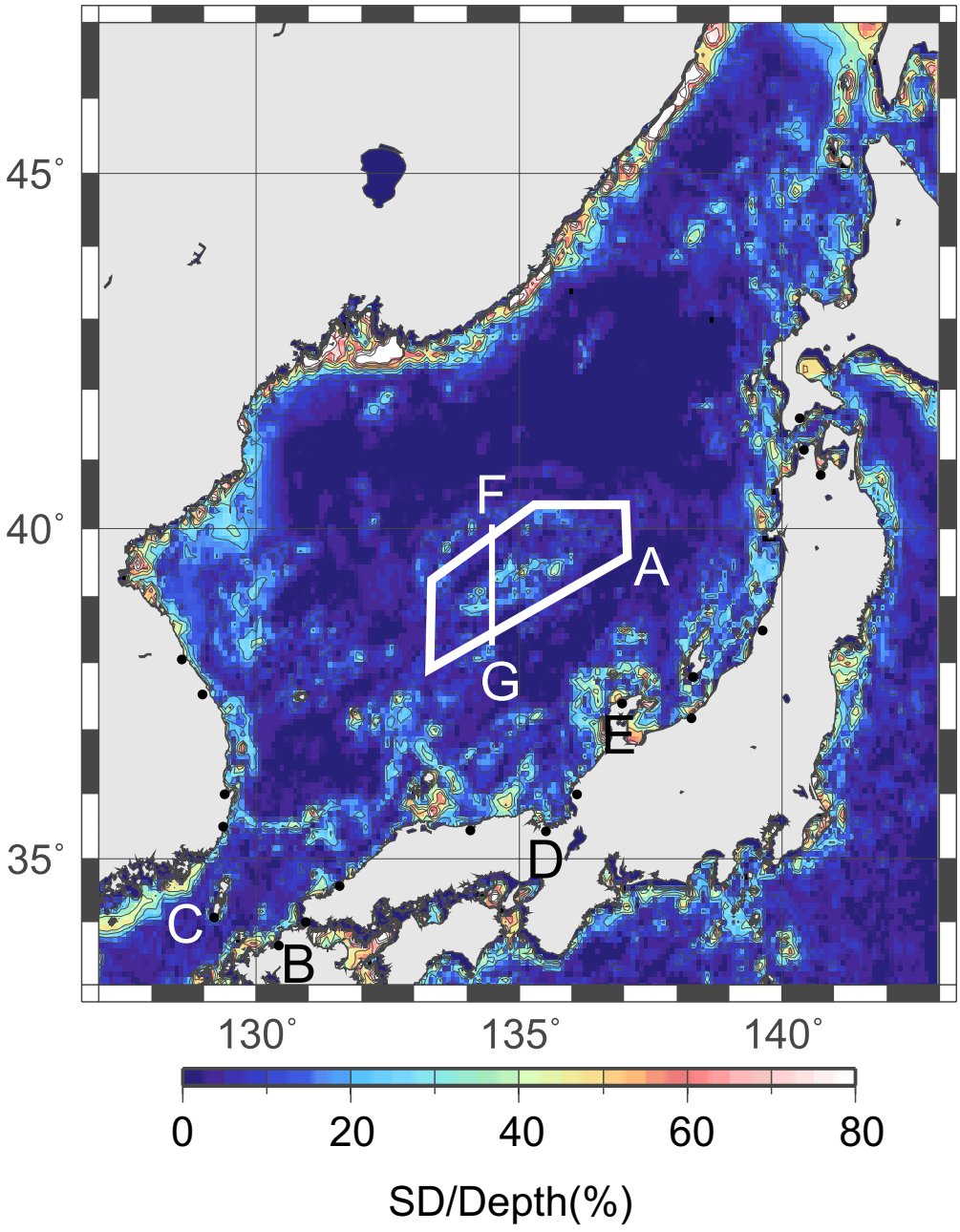

Figure 5.3: Area of tsunami simulation. Contour map shows the ratio of the standard deviation among the data sets to depth at each point, represented by the contour map. The installation points of used tide gauges are shown by dots. Because area A (white pentagon area) has large errors, it is modified by data assimilation.

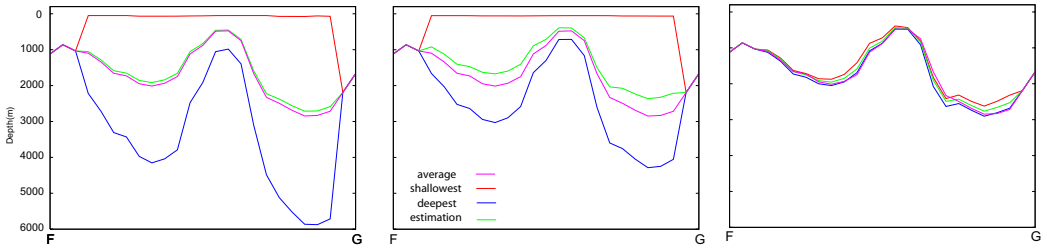

Figure 5.4: Snapshots of bottom topography profiles along segment FG in Figure 5.3.

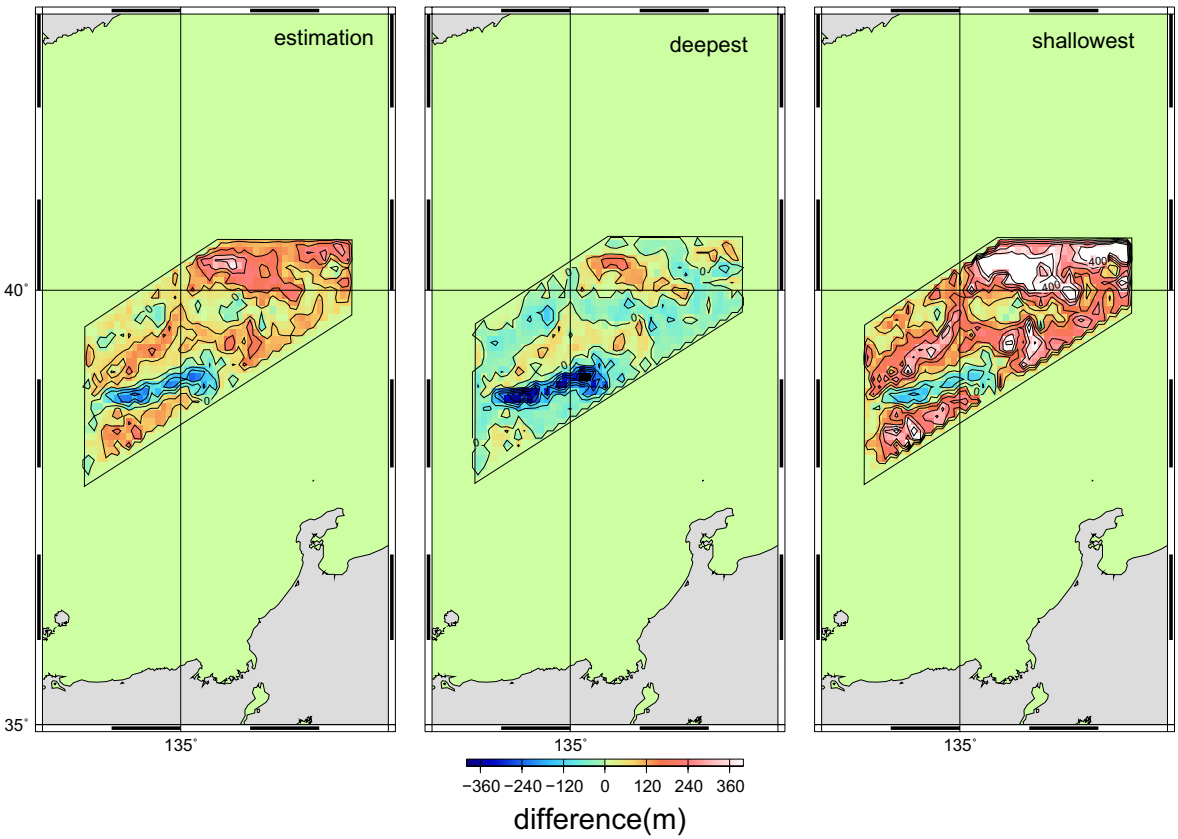

Figure 5.5: Contour plots of the differences between initial average and estimations.

5.3 Arrival Time Prediction

To validate the correction and prediction ability of the method, a simulation experiment of the Nihonkai–Chubu Earthquake Tsunami in 1983 was conducted. In this experiment, the estimated bottom topography obtained from the previous experiment and the original average of the available bottom topographies are used to simulate the tsunami and the results are compared with measured tide gauge data, with a particular focus on the arrival time. Table 5.1 shows a comparison of the tsunami arrival time at Hakata (point B in Figure 5.3), Izuhara (C) and Maizuru (D). Large difference of arrival time between observation and simulation in Hakata derives from unmodeled response with local topography, which implies simulation model can be improved further. Nevertheless, it can be seen that the simulation predictions are improved at some degree. Hakata and Izuhara are at the opposite side of the tsunami source from the Yamato Rises and the wave arrives at them directly. Hence the correction of the bottom topography gives a direct improvement. On the other hand, the wave which arrives at Maizuru through the assimilated area is not direct, but rather is a refracted wave. Therefore, the arrival time of the later refracted wave is modified. These results show that correction of bottom topography can give improvements of arrival time.

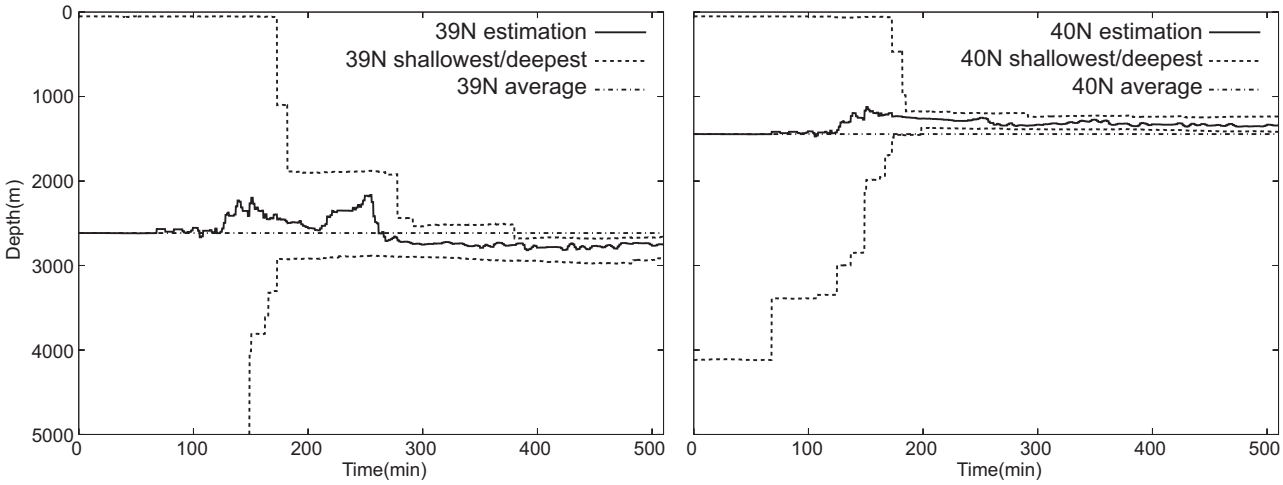

Figure 5.6: Estimated time series of the bottom topography at the points 39°N, 135°E and 40°N, 135°E.

Table 5.1: Arrival time from tsunami occurrence.

	observed time	unmodified	modified
	Hakata		
First peak	5h34m	3h52m	3h52m
Second peak	5h42m	4h24m	4h25m
	Izuhara		
First peak	3h42m	3h14m	3h15m
Second peak	3h52m	3h38m	3h39m
	Maizuru		
First peak	2h4m	1h47m	1h47m
Second peak	2h16m	2h8m	2h8m
Seventh peak (maximal)	3h40m	3h36m	3h38m

Chapter 6

Discussion

I have shown that the framework of the application of sequential data assimilation for the tsunami simulation model and tide gauge data. In this framework, the particle filter (PF) is used for estimation.

The following findings are obtained. At first, the identical twin experiment shows the applicability of the framework for real data. That is to say, sequential data assimilation for a tsunami simulation model and tide gauge data can be used to correct bottom topography. Secondly, the state estimation works well for a small number of ensemble members, in spite of the large dimensionality of the simulation model and the sparseness of the observations. This is allowed by the simple parametrization of uncertainties.

Thirdly, I have corrected bottom topography in and around the Yamato Rises and obtained that most points are shallower than the average of available data sets except for south slope of the South Rise. To confirm this suggestion rigorously, more measurements of the sea bottom are desired. Fourth point is that simulation of the Nihonkai–Chubu Earthquake Tsunami demonstrates the effectiveness and weakness of the method for tsunami prediction. The uncertainties of bottom topography account for the model error of the simulation, and thereby the prediction can be more precise whether obtained bottom topography is near to the true one. Though we can the improve the simulation and observation models in some respects, the result shows that the method can reduce error in tsunami prediction. To realize more precise prediction, coast area assimilation could be also used effectively

to modify models because the coast areas are also erroneous and shallower than the rises.

Part II

Toward High Dimensional Data Assimilation

Chapter 7

Introduction

7.1 Motivation

Data assimilation (DA) deals with high dimensional simulation models and needs to manage the difficulty derived from high dimensionality and nonlinearity. As I noted in Chapter 2, DA can be formulated as a state estimation problem by a nonlinear state space model (SSM). In this formulation, a simulation model is embedded into SSM as an evolutional model which makes up system model in SSM. The DA problem is an inverse problem in that there is less information in the observation data than the estimated variables. However, there are many differences between the DA and other fields using SSM, which make the estimation problem difficult. Table 7.1 represents some typical differences between the DA and other fields. Difficulties with DA are that the scale of the system model is extremely large compared with other fields using an SSM, the simulation model which is based on physical model is often given only by source code and computational cost to solve the simulation model is often extremely high. These difficulties should be resolved in the applications of DA.

In the 4DVAR [Talagrand and Courtier, 1987; Courtier *et al.*, 1994*a*,*b*] which is a batch type DA procedure, some properties enable us to reduce computational costs. At first, cost function depends only on the initial state vector. Secondly, it is quadratic form of state vectors under fixed time. On

the other hand, because the cost function is optimized under these conditions, it can easily fall into a local minimum. The ensemble based methods can partially avoid that problem if appropriate initial state vectors are set. However, the ensemble based methods need much more memory spaces to conserve state variables because estimation of state variable is represented by the set of samples. This means that the solvable dimension of state vector with the ensemble filters is smaller than that with other approaches.

In the PF community, it has been longed to resolve the problem known as degeneracy. It derives from limitations of procedure to keep good properties of approximated PDF. Applying smooth bootstrap to the PF is one of the candidates to avoid this degeneracy problem [Stavropoulos and Titterington, 2001]. This approach can keep good property that the estimated PDF converges in appropriate one as sample size goes infinity. However, it needs more computational cost than normal PF. Similar approach to this is the ensemble Kalman filter (EnKF, [Evensen, 1994, 2003]) used in the geophysics. The EnKF also avoids degeneracy problem. However, as I show in the following, the EnKF can not appropriately estimate the states and the parameters even in a small state vector. This problem arises from the assumption of Gaussianity of PDFs. Some variants of the EnKF such as ETKF (ensemble transform Kalman filter, [Bishop *et al.*, 2001; Majumdar *et al.*, 2001]), EAKF (ensemble adjustment Kalman filter, [Anderson, 2001]) and SEEK (Singular Evolutive Extended Kalman) filter [Pham *et al.*, 1998; Pham, 2001] have been proposed, but they basically adjust at most second order statistics and hence the problem was not resolved. Pham [2001] also proposed the PF with the smooth bootstrap approach in context of DA and researched the case of Lorenz 63 model. However, it concentrated on the case of simple observations and the states of the system are not difficult to determine, and then further researches are required such as the parameter identification with nonlinear observation models.

The relationship between the PF and genetic algorithms (GA) was referred by Higuchi [1996, 1997] and Iba [2001]. As noted in Iba [2001], the PF aims at calculating marginal PDF or ex-

pectation values whereas genetic algorithms aim at only optimization. The PF can be modified for optimization, but the converse of it, *i.e.*, modification of genetic algorithms to assess marginal PDF, is not necessarily evident. This problem derives from crossover operations in genetic algorithms. The EnKF conducts crossover procedure using the rule of Kalman filter, which achieves a good balance between crossover procedure and second order consistency. That is, an ensemble obtained by EnKF is a sample set from the PDF which is "true" if it is Gaussian. The limitation enables the crossover operation to conserve second order consistency. In the following, I will clarify the similarities and properties among ensemble filters and GA and locate the smooth bootstrap approach to the PF in them. It will give a new viewpoint about ensemble filtering, especially in DA, The motivation of the studies in Chapter 8 and 9 derives from this.

Whereas a filter gives an estimation of state vector at the time that the newest observation is obtained, a smoother gives the estimation of the past time step. Though smoothers are more suitable to the goals in some cases, applications are limited because of spatial and computational cost and the properties of smoothers. The ensemble Kalman smoother (EnKS, [Evensen and van Leeuwen, 2000; van Leeuwen, 2001; Evensen, 2003]) and the particle smoother (PS, [Kitagawa, 1996; Clapp and Godsill, 1999; Kitagawa and Sato, 2001]) is the ensemble based fixed-lag smoother (also applicable as fixed-interval smoother) derived from the EnKF and the PF. They can be used in DA theoretically, application to large scale simulation system is also limited. One of the reason is the heavy use of memory space. To overcome this difficulty, I introduced recursive recomputation approach to the PS. This can lessen the burden in memory space. In Chapter 10, I explain it and give experimental results through the analysis of Nikkei 225 stock index data.

7.2 Organization of Part II

In Chapter 8, the EnKF is reviewed. In next, application of smooth bootstrap to the PF are shown. I will also show and discuss the relation ship among the PF, the EnKF, smooth bootstrap based filter

and the GA. In Chapter 9, numerical experiments to compare the ensemble filters are shown. In Chapter 10, the recursive recomputation approach to the PS are shown. In this chapter, numerical experiment using Nikkei 225 stock index time series and nonlinear non-Gaussian SSM is conducted. In Chapter 11, I will sum up and discuss these approaches from the viewpoint of DA.

Table 7.1: Difference of characteristics between DA and other fields.

	DA	other fields
State vector dimension	10^3–10^6	10^0 – 10^2
Observation vector dimension	10–10^4	10^0 – 10^2
Evolutional model	physical	statistical
System representation	source code	analytic form
Computational cost	high	low

Chapter 8

Characterization of Ensemble Filters

8.1 Ensemble Kalman Filter

At first, I assume that the observation model is linear one (2.4), *i.e.*,

$$\boldsymbol{y}_t = H_t \boldsymbol{x}_t + \boldsymbol{w}_t.$$

The update rule of the EnKF from the prediction is as follows. At first, sample mean $\hat{\boldsymbol{x}}_{t|t-1}$ and covariance matrix $\hat{V}_{t|t-1}$ are calculated:

$$\hat{\boldsymbol{x}}_{t|t-1} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_{t|t-1}^{(i)},$$
$$\hat{V}_{t|t-1} = \frac{1}{N-1}\sum_{i=1}^{N} (\boldsymbol{x}_{t|t-1}^{(i)} - \hat{\boldsymbol{x}}_{t|t-1})(\boldsymbol{x}_{t|t-1}^{(i)} - \hat{\boldsymbol{x}}_{t|t-1})^{\mathrm{T}}$$

where $\cdot^{\mathrm{T}}$ denotes transpose of matrix. Filtered ensemble $\{\boldsymbol{x}_{t|t}^{(i)}\}_{i=1}^{N}$ is calculated through the update equation of the Kalman filtering

$$K_t = \hat{V}_{t|t-1} H_t^{\mathrm{T}} (H_t \hat{V}_{t|t-1} H_t^{\mathrm{T}} + \hat{R}_t)^{-1},$$
$$\boldsymbol{x}_{t|t}^{(i)} = \boldsymbol{x}_{t|t-1}^{(i)} + K_t(\boldsymbol{y}_t + \boldsymbol{w}_t^{(i)} - H_t \boldsymbol{x}_{t|t-1}^{(i)})$$

where $\boldsymbol{w}_t^{(i)}$ denotes sample from $N(\boldsymbol{0}, R_t)$ and $\hat{R}_t$ denotes the sample covariance matrix of $\boldsymbol{w}_t^{(i)}$. It is important to note that this procedure assumes Gaussianity of $\boldsymbol{x}_{t|t-1}$ and linearity in the observation model.

Because filtering step of the EnKF are constructed by the filtering step of Kalman filter, the EnKF cannot deal with nonlinear observation system

$$\boldsymbol{y}_n = h_n(\boldsymbol{x}_n) + \boldsymbol{w}_n. \tag{8.1}$$

State vector extension technique is used by Evensen [2003] to avoid this problem. At first, state vector $\boldsymbol{x}_t$ is extended to $\tilde{\boldsymbol{x}}_t$ in this technique:

$$\tilde{\boldsymbol{x}}_t = [\boldsymbol{x}_t^{\mathrm{T}}, h_t(\boldsymbol{x}_t)^{\mathrm{T}}]^{\mathrm{T}}.$$

Next, formally linear observation model is introduced:

$$\boldsymbol{y}_t = \tilde{H}_t \tilde{\boldsymbol{x}}_t + \boldsymbol{w}_t,$$

where $\tilde{H}_t = [O_{n_y \times n_x}, I_{n_y \times n_y}]$. This formally linear model is used instead of (8.1).

8.2 Matrix Representation

In the context of the EnKF, the filtering step is written by matrix representation of Evensen [2003]. All the ensemble members are included in the matrix as column vectors. For example, the ensemble $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^N$ is written by

$$X_{t|t-1} = [\boldsymbol{x}_{t|t-1}^{(1)}, \boldsymbol{x}_{t|t-1}^{(2)}, \cdots, \boldsymbol{x}_{t|t-1}^{(N)}].$$

The update rule of the EnKF can be written through this representation. Evensen showed that the update rule can be written by

$$X_{t|t} = X_{t|t-1} Z_{t|t-1}, \tag{8.2}$$

where $Z_{t|t-1}$ is calculated from $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^N$, $\{\boldsymbol{w}_t^{(i)}\}_{i=1}^N$, H_t and $\boldsymbol{y}_t$ [Evensen, 2003]. It is also shown that the sum of each column of $Z_{t|t-1}$ is one. These things show that the filtered ensemble members are weighted linear combination of the predictive ensemble members.

Once the matrix representation of the ensemble is introduced, the PF can be written by the same way. Resample from $\{\boldsymbol{x}_{t|t-1}^{(i)}\}_{i=1}^{N}$ is written by

$$X_{t|t} = X_{t|t-1}Z_{t|t-1},$$

where each element of $Z_{t|t-1}$ takes the value of zero or one. In this formulation, each column of $Z_{t|t-1}$ consists of a one and $N-1$ zeros. Additionally, the number of ones which appear in the ith row of $Z_{t|t-1}$ is proportional to the weight of ith member of prediction ensemble $\boldsymbol{x}_{t|t-1}^{(i)}$.

This representation of filters gives us the consistent view of the EnKF and the PF. The rank of $Z_{t|t-1}$ shows the non-degeneracy of the ensemble, which means decay of variation of the ensemble members. Degeneracy may cause severe estimation bias in the sequential DA because predictive ensemble members which are generated from the same filtered ensemble member may have almost the same value and physical characteristics. Because $Z_{t|t-1}$ of the EnKF is usually full rank, degeneracy of the ensemble $\{\boldsymbol{x}_{t|t}^{(i)}\}_{i=1}^{N}$ rarely occurs. In addition, predictive ensemble members span the same space as filtered ensemble members do. This property is desirable if nonlinearity of system model (2.2) is weak from the viewpoint of physical simulation model. However, the filtered ensemble of the EnKF can assure the accuracy of moment only up to second order, nonlinearity of SSM may cause estimation errors. On the other hand, $Z_{t|t-1}$ of the PF is rarely full rank because the PF is resample based method and the number of ensemble members are finite. This rank deficiency is cause of degeneracy and estimation bias in the PF though the PF can assure accuracy of higher order statistics.

As a consequence of the properties, if the $Z_{t|t-1}$ is efficiently calculated without loss of rank and statistical bias, estimated ensembles will be more efficient and accurate estimation.

8.3 Smooth Bootstrap Based Filtering

As noted in the previous section, the PF makes filtered samples from finite sample set, which causes degeneracy problem. The same problem exists in the bootstrap technique [Efron and Tibshirani,

1993]. Smooth bootstrap is a variant of the bootstrap technique. While bootstrap method makes finite sample set from itself, smooth bootstrap samples from the value around sample set which enables us to avoid the degeneracy. Therefore, application of smooth bootstrap to the PF (in the following, the smooth PF) might avoid degeneracy problem [Stavropoulos and Titterington, 1998, 2001]. According to the proof by Stavropoulos and Titterington [1998], desirable properties of the PF such as consistency are preserved under some conditions of smoothing rule. Gaussian function is used as a smoothing kernel function in many cases. To implement the smooth PF, we should only replace resampling procedure with the following:

- Determine $\{\boldsymbol{x}_{t|t}^{(i)}\}_{i=1}^{N}$ by sampling from the PDF $\sum_{i=1}^{N} q_t^{(i)} N(\boldsymbol{x}_{t|t-1}^{(i)}, b_N^2 \hat{V}_{t|t-1})$,

Here, $\hat{V}_{t|t-1}$ is sample covariance matrix calculated from $\{\boldsymbol{x}_{t|t-1}^{(i)}\}$ and b_N^2 is a non-negative scalar function of N which suffices

$$b_N^2 \longrightarrow 0. \tag{8.3}$$

To make the smooth PF work correctly, this condition is required [Stavropoulos and Titterington, 2001] since the variance of the smoothing kernel function should vanish as sample size goes infinity.

When the state vector is multi-dimensional, sample covariance matrix of state vector $\boldsymbol{x}_{t|t-1}$ represents the direction of sample distribution.

8.4 Relationship to Genetic Algorithm

In this section, I will give correspondence and difference between the ensemble based filters and Genetic algorithms (GA). At first, I should point out that GA originally aims at solving optimization problem while the ensemble based approach aims at obtaining PDF or related statistics.

8.4.1 Mutation and Growth

In the PF, the procedure similar to mutation in GA is the part of introducing noise $\boldsymbol{v}_t$ in state vector $\boldsymbol{x}_t$ by system model (2.2) except for some special cases. This is common in all ensemble based filters as noted in this chapter. Determining mutation rule corresponds to modeling of equation (2.2) and covariance matrix Q_t.

On the other hand, growth in GA corresponds to the prediction step of ensemble based filters. Mutation step is also included in the prediction step, but if we bring

$$\boldsymbol{x}_t = f_t(\boldsymbol{x}_{t-1}, \boldsymbol{0})$$

to the system model, prediction step includes growth only.

8.4.2 Selection and Crossover

PF case

Relationship between filtering step of the PF and selection in GA is simple. Likelihood function $L(\boldsymbol{x}_t|\boldsymbol{y}_t)$ corresponds to an objective function in GA. Therefore, filtering step of the PF selects or duplicates more likely samples from the predicted sample set on the basis of likelihood as GA does in accordance with fitness. Crossover is not introduced in the PF. Existence of selection procedure and lack of crossover cause degeneracy problem in the PF. It corresponds to the problem in GA that the parameters fall into and cannot escape from local minimum when crossover is not implemented.

Smooth bootstrap case

The smooth PF has both selection and crossover type procedures in the filtering step. The larger the weight of ith sample $\boldsymbol{x}_{t|t-1}^{(i)}$ is, the more samples are generated around ith sample. As a consequence, sample may not be generated around less weight sample. This corresponds to selection.

It should be carefully examined for crossover effect, but if Gaussian function is adopted as a kernel function and sample covariance matrix is used for this, filtering procedure has both selection

and crossover. Figure 8.1 illustrates the smooth PF procedure and shows that crossover effect is included: Sample covariance matrix is calculated in (a); high probability range around each predicted sample is determined in (b); sampled from the distribution which is determined by update equation in (c) and filtered sample set is given in (d). Using Gaussian and sample covariance matrix limits sample generation direction, which means that filtered samples $\{\boldsymbol{x}_{t|t}^{(i)}\}$ can only be generated along linear span of $\{\boldsymbol{x}_{t|t-1}^{(i)}\}$.

The crossover is performed for all samples in this procedure. Therefore, occurrence possibility of degeneracy problem is smaller than the PF. Of course if b_N or $V_{t|t-1}$ is very small compared to the desirable dispersion of $\{\boldsymbol{x}_{t|t}^{(i)}\}$, situation that samples are approximately degenerated may occur, but at least degeneracy may hardly happens in the strict meaning.

Selection of kernel or parameter may introduce a mutation effect at the filtering step. For example, suppose the situation that $V_{t|t-1}$ is not full rank and used Gaussian kernel with full rank covariance matrix which is independent to data. Then generated filter samples $\{\boldsymbol{x}_{t|t}^{(i)}\}$ are not linear combination of $\{\boldsymbol{x}_{t|t-1}^{(i)}\}$ any more, *i.e.*, mutation is also introduced. Though that is not necessarily undesirable, avoiding these situation is a good choice for consistency with other techniques.

EnKF case

The filtering procedure of the EnKF mainly consists of crossover in GA, and selection is not conducted. To see this fact, I change update equation from $\{\boldsymbol{x}_{t|t-1}^{(i)}\}$ to $\{\boldsymbol{x}_{t|t}^{(i)}\}$ by the following:

$$\begin{aligned}\boldsymbol{x}_{t|t}^{(i)} =& \boldsymbol{x}_{t|t-1}^{(i)} + K_t(y_t - H_t\boldsymbol{x}_{t|t-1}^{(i)} + w_t^{(i)}) \\ =& (I - K_tH_t)\boldsymbol{x}_{t|t-1}^{(i)} + K_ty_t + K_tw_t^{(i)}. \end{aligned} \tag{8.4}$$

First term of (8.4) is linear transformation which is common in all samples. Second term is parallel translation which is also common. Last term is "jittering" term which moves each samples independently. Direction of jittering is determined independently among samples. Therefore, there is no multiplication or selection part.

Lack of selection leads to inefficiency of PDF estimation. That is, we have trouble removing a "bad" sample in estimation. On the other hand, degeneracy problem rarely occurs and stability of calculation can be easily conserved.

8.5 Characterization of Filters

I summarize the characterization of the PF, the EnKF and the smooth PF. Table 8.1 shows the summary. Because prediction step is common, we only need to compare filtering steps. The PF needs the least computational burden because matrix inverse is only needed for R_t and selection can be used efficient algorithm. The EnKF and the smooth PF follow this because sample covariance matrix is required. Because efficient calculation procedure is developed in the EnKF by using the properties specific to it [Evensen, 2003, 2004], it needs less computer resources than smooth PF does. Both of them need sample covariance matrix for the purpose of making support, but such calculation may fundamentally be unnecessary to make support only.

8.6 Ensemble Smoothers

Both the EnKF and the PF have corresponding smoothers, the ensemble Kalman smoother (EnKS, [Evensen and van Leeuwen, 2000; Evensen, 2003]) and the particle smoother (PS, also known as Monte Carlo smoother, [Kitagawa, 1996; Doucet *et al.*, 2000; Kitagawa and Sato, 2001; Godsill *et al.*, 2004]). The PS based on "storing the state vector" approach and the EnKS have similar procedure and can be represented by the same way using matrix representation.

The fixed L-lag smoother calculates distributions $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t+j})$, $(j = 0, \cdots, L-1)$. Ensemble based fixed L-lag smoothers can be obtained only by replacing $X_{t|t-1}$ in the filtering equation (8.2) with

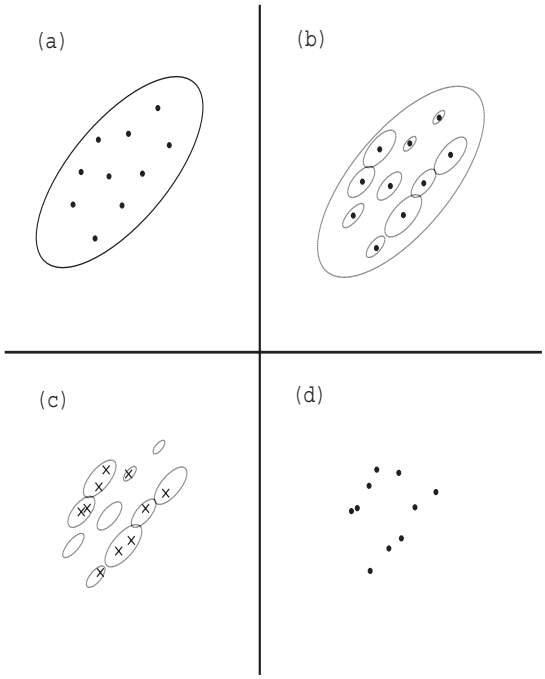

Figure 8.1: Illustration of smooth bootstrap in two dimensional case

Table 8.1: Summary table of three filters and GA

	PF	Smooth PF	EnKF	GA
Selection	√	√	none	√
Crossover type op.	none	√	√	√
PDF estimation	accurate	accurate	inaccurate	none
— efficiency	√	less	none	none
Degeneracy	usual	partial	rare	partial
Comp. Cost	low	highest	high	low

L-lag stored matrix

$$\Xi_{t|t-1} = \begin{pmatrix} \boldsymbol{x}_{t|t-1}^{(1)} & \boldsymbol{x}_{t|t-1}^{(2)} & \cdots & \boldsymbol{x}_{t|t-1}^{(N)} \\ \boldsymbol{x}_{t-1|t-1}^{(1)} & \boldsymbol{x}_{t-1|t-1}^{(2)} & \cdots & \boldsymbol{x}_{t-1|t-1}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{t-L+1|t-1}^{(1)} & \boldsymbol{x}_{t-L+1|t-1}^{(2)} & \cdots & \boldsymbol{x}_{t-L+1|t-1}^{(N)} \end{pmatrix}$$

where $\boldsymbol{x}_{t'|t^*}^{(i)}$ denotes smoothed ensemble members at time t' using $\boldsymbol{y}_{1:t^*}$. $Z_{t|t-1}$ is the same one in the equation (8.2). Then, we can obtain smoothed ensemble through the equation

$$\Xi_{t|t} = \Xi_{t|t-1} Z_{t|t-1}$$

where $\Xi_{t|t}$ denotes

$$\Xi_{t|t} = \begin{pmatrix} \boldsymbol{x}_{t|t}^{(1)} & \boldsymbol{x}_{t|t}^{(2)} & \cdots & \boldsymbol{x}_{t|t}^{(N)} \\ \boldsymbol{x}_{t-1|t}^{(1)} & \boldsymbol{x}_{t-1|t}^{(2)} & \cdots & \boldsymbol{x}_{t-1|t}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{t-L+1|t}^{(1)} & \boldsymbol{x}_{t-L+1|t}^{(2)} & \cdots & \boldsymbol{x}_{t-L+1|t}^{(N)} \end{pmatrix}.$$

Fixed interval smoother is similarly constructed by replacing the matrix $\Xi_{t|t-1}$ and $\Xi_{t|t}$ with $t+1$

$$\Xi'_{t|t-1} = \begin{pmatrix} \boldsymbol{x}_{t|t-1}^{(1)} & \boldsymbol{x}_{t|t-1}^{(2)} & \cdots & \boldsymbol{x}_{t|t-1}^{(N)} \\ \boldsymbol{x}_{t-1|t-1}^{(1)} & \boldsymbol{x}_{t-1|t-1}^{(2)} & \cdots & \boldsymbol{x}_{t-1|t-1}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{0|t-1}^{(1)} & \boldsymbol{x}_{0|t-1}^{(2)} & \cdots & \boldsymbol{x}_{0|t-1}^{(N)} \end{pmatrix} \text{ and } \Xi'_{t|t} = \begin{pmatrix} \boldsymbol{x}_{t|t}^{(1)} & \boldsymbol{x}_{t|t}^{(2)} & \cdots & \boldsymbol{x}_{t|t}^{(N)} \\ \boldsymbol{x}_{t-1|t}^{(1)} & \boldsymbol{x}_{t-1|t}^{(2)} & \cdots & \boldsymbol{x}_{t-1|t}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{x}_{0|t}^{(1)} & \boldsymbol{x}_{0|t}^{(2)} & \cdots & \boldsymbol{x}_{0|t}^{(N)} \end{pmatrix}.$$

Calculation cost of fixed interval smoother is $O(N^2T^2)$ if it is implemented naively. Particular implementations of resampling which corresponds to construction of $Z_{t|t-1}$ can reduce the cost from $O(N^2)$ to $O(N)$, and therefore, calculation cost of fixed interval smoother can be reduced to $O(NT^2)$. Moreover, all the informations about the parents of filtered particles are stored, calculation cost of fixed interval smoother is reduced to $O(NT)$ though the required memory is increased. This type of algorithm is applied at the numerical experiment in the following section.

Chapter 9

Numerical Experiment

I used two types of system models for numerical experiments, one dimensional bi-modal model [Carlin *et al.*, 1992; Gordon *et al.*, 1993; Kitagawa and Gersch, 1996; Kitagawa, 1998] and Lorenz 96 model [Lorenz and Emanuel, 1998]. The former model is widely used in the nonlinear identification problem in statistics and signal processing. The latter is used as a testbed in geophysics.

9.1 Experiment on Highly Nonlinear Observation System

I used the simulation data y_t that were generated by the following system [Kitagawa, 1998]:

$$x_t = \frac{1}{2}x_{t-1} + \frac{25x_{t-1}}{1 + x_{t-1}^2} + 8\cos(1.2t) + v_t, \tag{9.1}$$

$$y_t = \frac{x_t^2}{20} + w_t \tag{9.2}$$

where $x_0 \sim N(0, 5)$, $v_t \sim N(0, 1)$, $w_t \sim N(0, 10)$ and $t = 1, \cdots, T$. The reason why I used this system is that it is widely used in the statistical field and its properties are known very well [e.g., Doucet *et al.*, 2001]. I researched the performance of the filters and the smoothers by the following three experiments.

9.1.1 Performance Comparison of the EnKF/EnKS and the PF/PS

In the first experiment, the model (9.1) and (9.2) was used for estimation of the state x_t, and the state was estimated with the PF, the EnKF, the PS and the EnKS. Nonlinear observation was processed according to the method in the previous section. Lag length of the PS and the EnKS is set to be 20. I used the average of ensemble members as estimation variable. In the second experiment, the variance of the system noise v_t was treated as unknown parameter σ_v to be estimated and was estimated together with state. The estimation is based on the self-organizing state-space model [Kitagawa, 1998] in which the system model (9.1) is replaced by

$$\begin{pmatrix} x_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} \frac{1}{2}x_{t-1} + \frac{25x_{t-1}}{1+x_{t-1}^2} + 8\cos(1.2t) \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} v_t \\ u_t \end{pmatrix}, \tag{9.3}$$

$$x_0 \sim N(0, 5),\ v_t \sim N(0, \exp(\theta_t)),\ u_t \sim N(0, \xi), \tag{9.4}$$

where ξ is predetermined constant. In this model, the estimated variance σ_v should be $\exp(\theta_t)$. The estimations by the PF and the EnKF were compared with the estimation by the numerical integration [Kitagawa, 1987] as most accurate estimation. Because the dimension of this system is small, we can use integration. In the real application, the curse of dimension prevents us from using this technique.

State Estimation

Figure 9.1 shows the observation series y_t and Figure 9.2 shows corresponding state series x_t and estimated state series $\hat{x}_t$ by the EnKF and the PF. it is seen that estimation by the PF is more accurate than by the EnKF.

Table 9.1 shows the average of the sum of squared difference $\sum_1^T (\hat{x}_n - x_n)^2$ among 100 experiments. Estimation experiments were done 100 times under the condition that the number of ensemble members is 100, 1000 and 2500. It can be seen that estimation by the PF is more accurate than by the EnKF if the number of ensemble members is same. The same relationship between the PS and the EnKS can be seen.

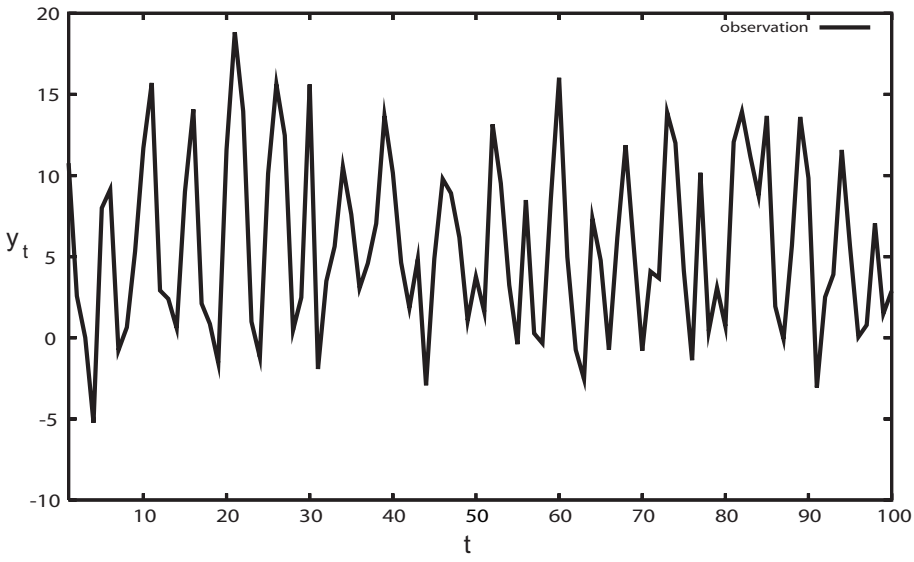

Figure 9.1: Example of observation series. Vertical axis represents observation y_t and horizontal axis represents time t.

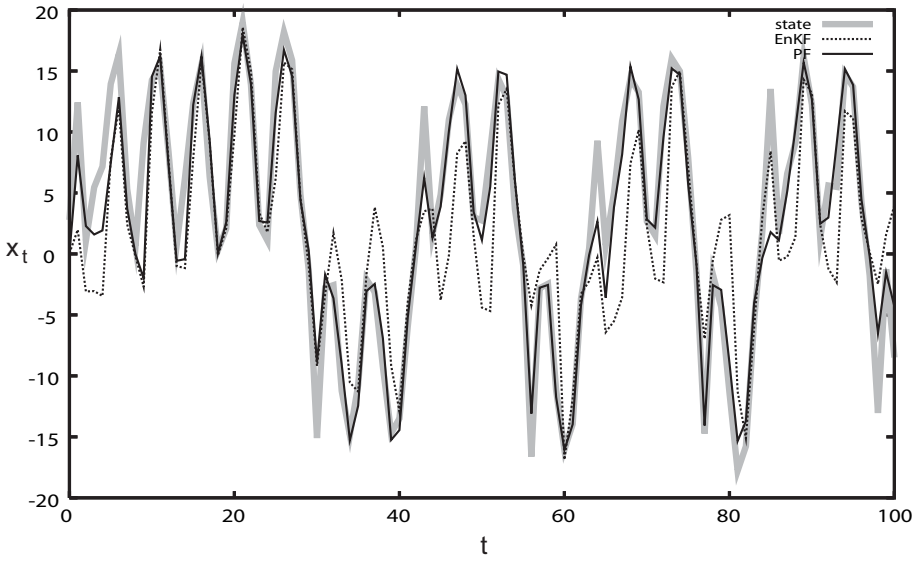

Figure 9.2: State series corresponding to observation series in Figure 9.1 and estimated state series by the EnKF and the PF. Vertical axis represents state x_t and horizontal axis represents time t.

These results show the superiority of the PF to the EnKF in terms of the accuracy of the state estimation. The superiority of the PS to the EnKS is also demonstrated. On the other hand, if we apply the PF to real data assimilation, degeneracy problem should be resolved. In this numerical experiments, because the dimension of state vector n_x is 1, degeneracy problem is covered by the richness of the number of the ensemble members compared to the dimension of x_t.

Parameter Identification

Figure 9.3 shows the result of the parameter estimation. We can regard the result of numerical integration as accurate estimation. In this experiment, the number of the ensemble members is 100. This result also shows the accuracy of parameter estimation by the PF.

9.1.2 Performance Comparison Among the smooth PF and Other Filters

To compare the smooth PF and other filters, I performed numerical experiments using the smooth PF and other filters under almost same settings above.

State Estimation

Figure 9.4 shows an example of estimation result of PDF of state. The number of the samples is 100. The figure shows the snapshot of the distribution of state samples at time $t = 100$. The true value of state is $x_{100} = -1.069$. Procedure of Smooth PF/independent in Figure 9.4 is variant of the smooth PF. Changed point from original smooth PF is the setting of covariance estimation where correlations of state variables are set to zero. Estimating PDF of the state, there is no distinct difference between the PF and the smooth PF. On the other hand, the EnKF is conservative and so the estimation performance is worse. These are expected results from the nature of these filters. The reason why degeneracy does not occur in the PF is that the “mutation” ratio is high and dimension of state is low. These are confirmed in the next experiment.

Table 9.1: The sum of squared differences. Estimation experiments were done 100 times and the average of the sums was calculated.

Number of ensembles	PF	EnKF	PS	EnKS
100	1841.76	2853.11	567.84	1618.73
1000	1710.01	2779.09	404.90	1470.93
2500	1701.90	2771.28	397.03	1447.65

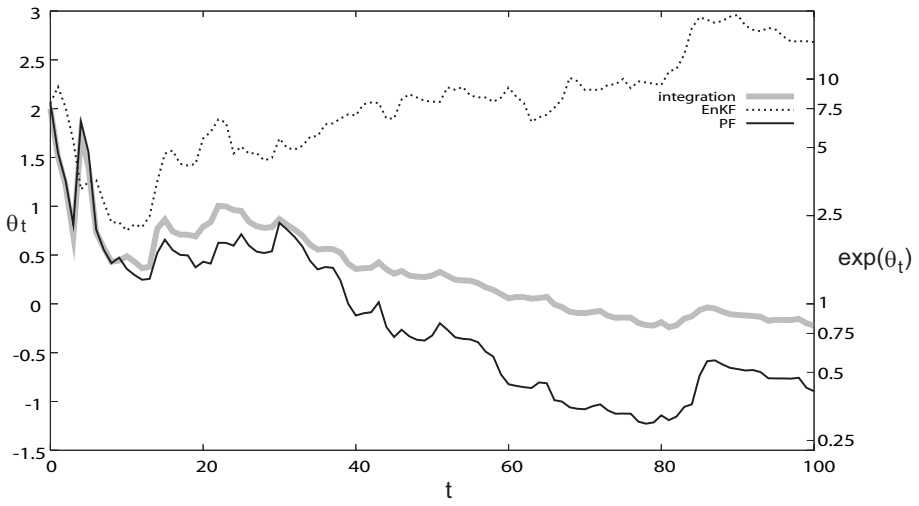

Figure 9.3: Parameter estimation using the EnKF and the PF. Left vertical axis represents θ_t, right vertical axis represents $\exp(\theta_t)$. Horizontal axis represents time t.

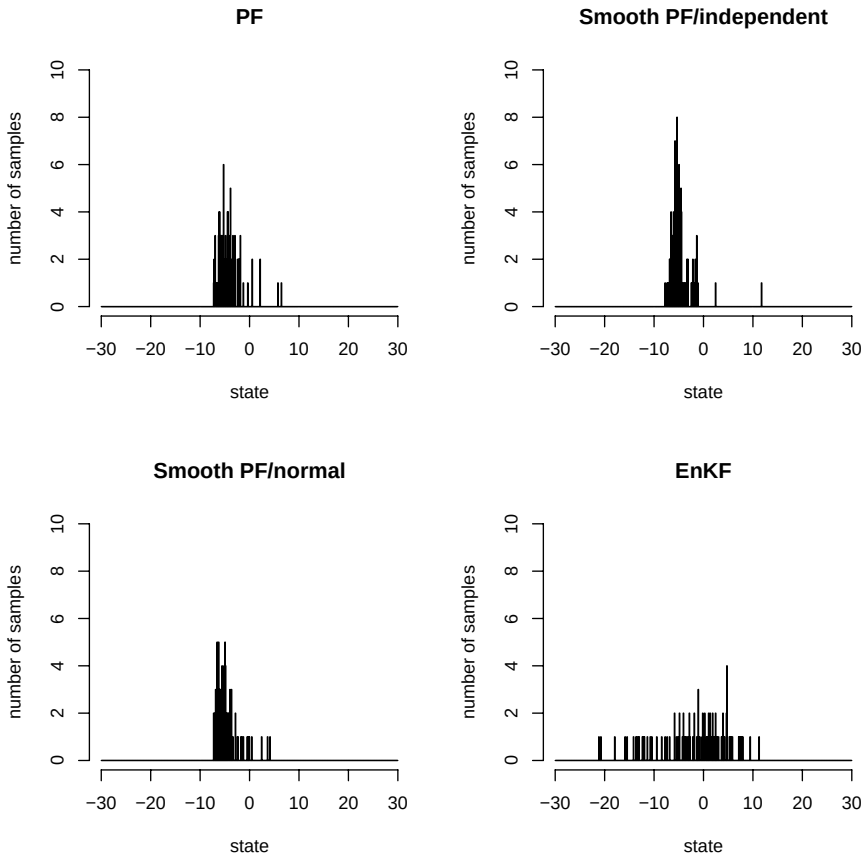

Figure 9.4: State estimation result at time $t = 100$

Parameter Identification

System variance as time invariant parameter by self-organizing state space model is estimated. The estimation is given by the filtering distribution at the last time step, *i.e.*, at the time T. Figure 9.5 (θ_t is disturbed) and Figure 9.6 (θ_t is not disturbed, *i.e.*, $xi = 0$) shows the result of estimation. Each of them represents histogram of $\{\theta_T^{(i)}\}$. Range of each bin is 0.1.

The result of figures shows the superiority of the smooth PF, in that the estimated value is near to true value zero and degeneracy does not occur. Degeneracy occurs in the PF in spite of introducing system noise. From the viewpoint of estimation, it is better if θ_t is not disturbed. In the estimation by the smooth PF, stable estimation is achieved in both figures and better in Figure 9.6, which is desirable. On the other hand, the result of estimation by the EnKF is poor in the parameter estimation too. The reason is the same in the state estimation case.

9.2 Numerical Experiment on Lorenz 96 Model

In this subsection, I compare the performance of state estimation by the EnKF, the PF and the smooth PF in the Lorenz 96 model [Lorenz and Emanuel, 1998].

9.2.1 Lorenz 96 Model

The fundamental differential equation of Lorenz 96 model is as follows:

$$\frac{\mathrm{d}\alpha_s}{\mathrm{d}t} = \alpha_{s-1}(\alpha_{s+1} - \alpha_{s-2}) - \alpha_s + F, \tag{9.5}$$

where F=8.0, $s = 1, 2, \cdots, 40$ and boundary condition is cyclic, *i.e.*, $\alpha_{-1} = \alpha_{39}$, $\alpha_0 = \alpha_{40}$ and $\alpha_{41} = \alpha_1$. This ODE is discretized by fourth order Runge-Kutta scheme [*e.g.*, Press *et al.*, 1992] with time step 0.05 and that is set to simulation model. Figure 9.7 shows a simulation result. Advection structure and irregularity in the Lorenz 96 model can be confirmed.

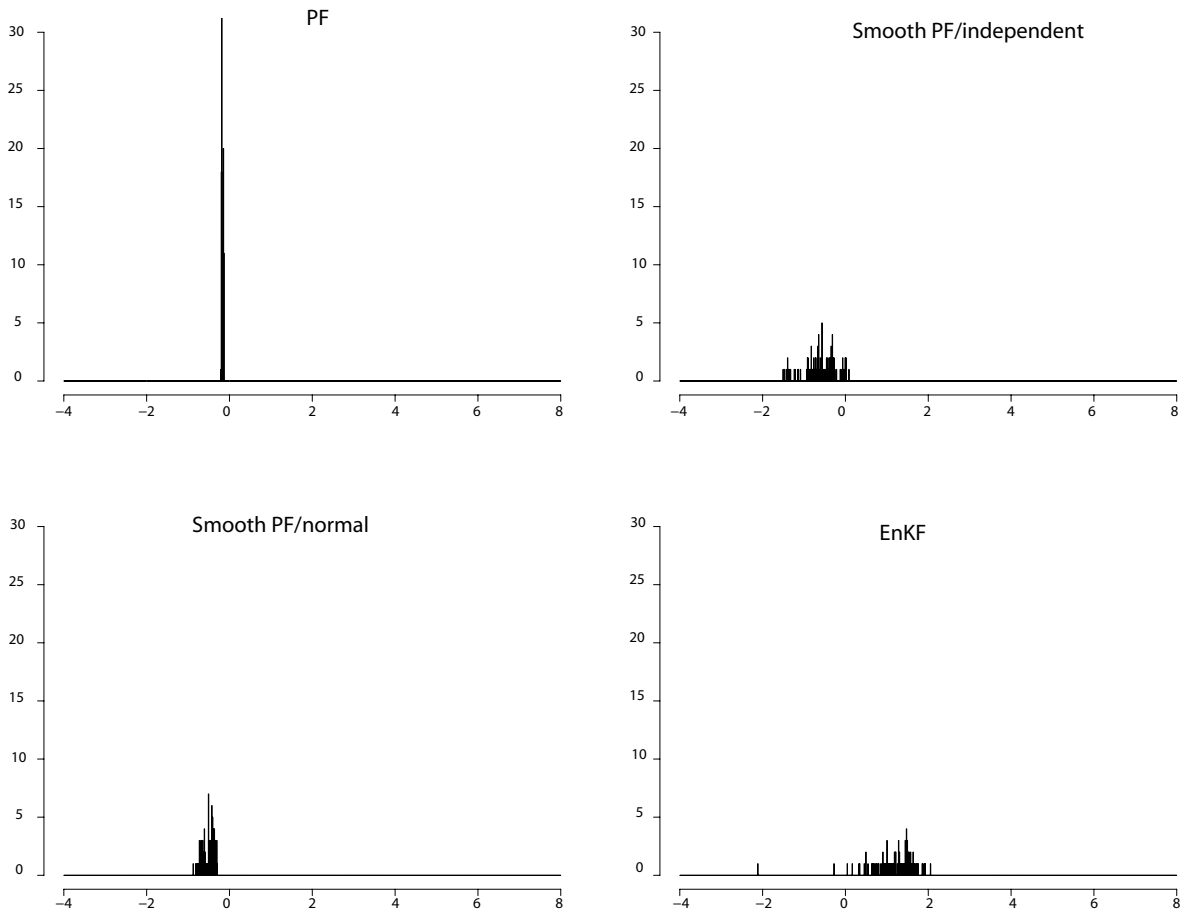

Figure 9.5: Parameter estimation; parameter is disturbed by system noise ($\xi = 0.05^2$).

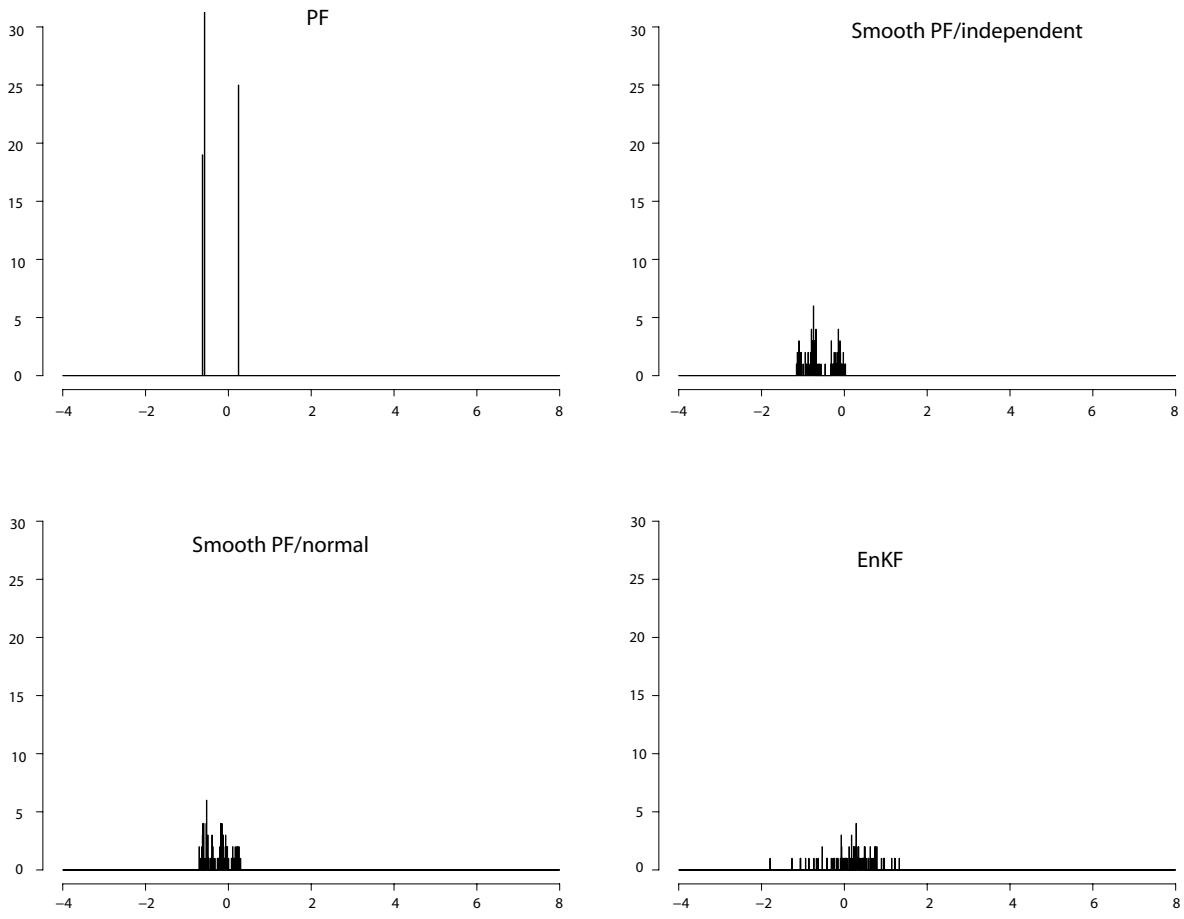

Figure 9.6: Parameter estimation; parameter is not disturbed.

It can be easily imagined that it is chaotic system and has long-term non-predictability because of non-linearity though it is predictable in short term. Figure 9.8 shows these properties. It is simulation result made by two initial conditions. The difference between them is only addition of 1.0×10^{-5} at α_1. The number of positive Lyapunov exponents is 13 [Ott *et al.*, 2004]. It also indicates instability of this system.

9.2.2 Numerical Experiment in Lorenz 96 Model

In this numerical experiment, I used random forcing, random observation setting. Forcing F was set to random variable whose distribution was $N(8.0, 0.01^2)$. This part was regarded as system noise. Observation points were set randomly in spatial. Each observation was obtained by linear combination of two nearest neighbor of the observation point. The coefficient of the linear combination was in inverse ratio to the distances. The variance of the observation noise was 1.0. The number of observation points were set to 10 and the number of particles were set to 100, 200 and 500. System and observation models were naturally introduced in the previous settings. First 1000 steps are used for free run and next 3000 steps are used for assimilation. RMS of differences between simulation states and assimilation states in the last 1000 steps are calculated. Table 9.2 shows the result.

The following things are observed in the table. The EnKF is better than the PF and the smooth PF in this setting. On the other hand, the errors reduce as the number of particles increases in the PF and the smooth PF. The reason of them might derive from the linearity of the observation system. Because observation system is linear, filtered distribution might be unimodal and the EnKF can obtain more stable estimation. It is also followed by the reduction of the errors in the PF and the smooth PF.

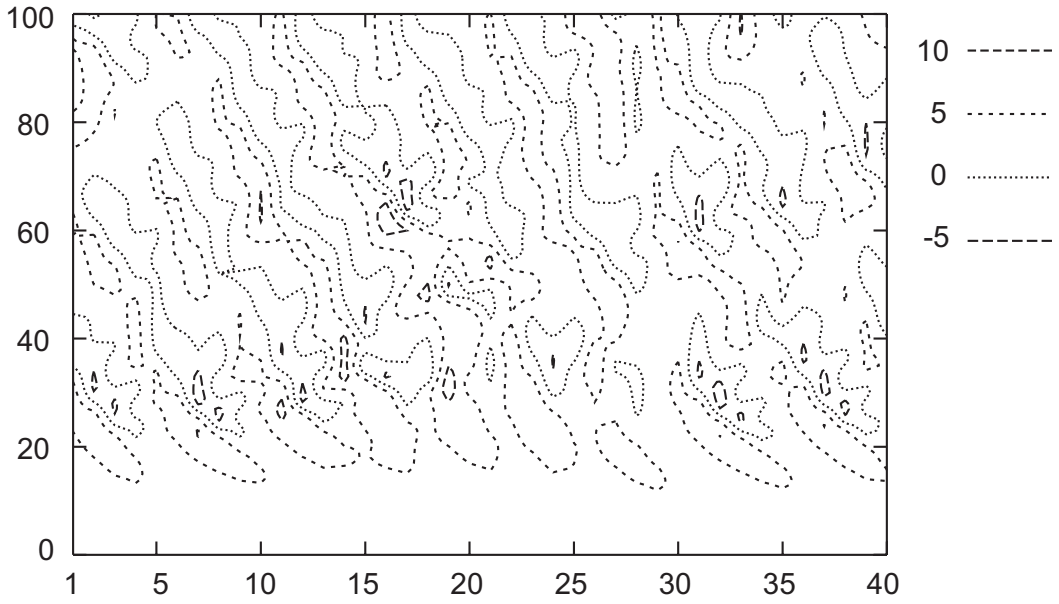

Figure 9.7: Contour plot of Lorenz 96 model. Horizontal axis means site s and vertical axis means time step. As time elapse, high or low value moves from right to left.

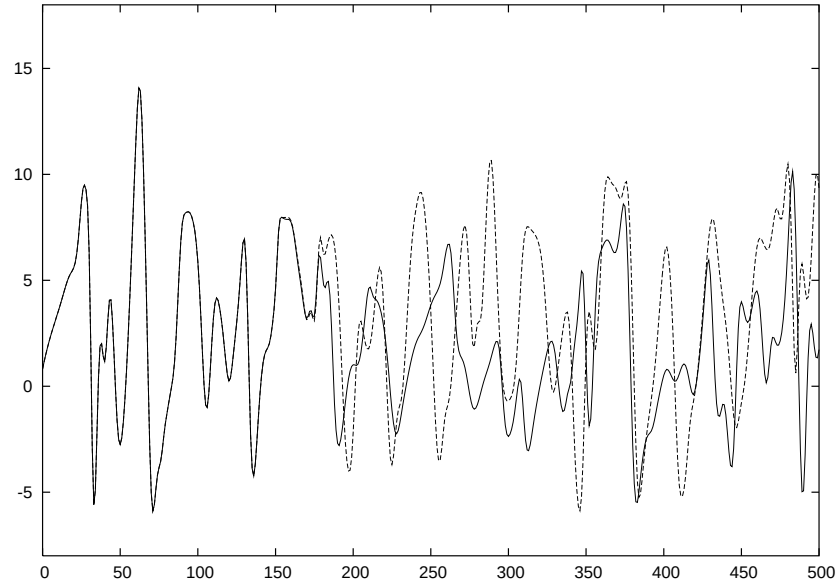

Figure 9.8: Time series of α_{15}. The difference of initial condition between solid and dashed line is only 1.0×10^{-5} at α_1.

Table 9.2: The sum of squared differences. The number of particles is 100.

number of particles	PF	smooth PF	EnKF
100	0.1700	0.1980	0.1232
200	0.1528	0.1710	0.1277
500	0.1489	0.1687	0.1298

Chapter 10

Recursive Recomputation Approach in Smoothing

Recursive recomputation technique is demonstrated in the following, which is another approach to resolving high dimensionality problem. As I noted in Section 8.6, the degeneracy problem is more critical in the PS rather than in the PF and hence the interval capable to be smoothed is limited. In Section 10.1, I explain recursive recomputation technique which can reduce the space complexity from $O(MT)$ to $O(M \log T)$ in fixed-interval smoother, where M is the memory space to store the sample set of state vector at each time step and T is the the number of time steps ($O(M)$ is proportional to $O(N)$). This approach can lessen the burden to store samples and hence longer time series or larger state vector can be smoothed. In Section 10.2, the result of numerical experiments on volatility estimation of Nikkei 225 stock index data is shown. Non-linear non-Gaussian state space model is employed and showed and smoothing in longer time scale can be achieved. The properties of the new scheme is also shown.

10.1 Recursive Recomputation

In this section, I explain application of the new scheme, "recursive recomputation scheme," to the PS. Though this approach is applicable for fixed-lag smoother [Nakamura and Tsuchiya, 2006], I con-

centrate on the fixed-interval smoother because it is enough to discuss the potential and applicability of the new scheme.

10.1.1 Recursive Recomputation Scheme

The new approach requires that all the generated random number sequence at some points of calculation can be recovered. Because major pseudo random number generators such as linear congruential generators [e.g., Press *et al.*, 1992] can recover them from the state variable and therefore it is applicable. Mersenne twister [Matsumoto and Nishimura, 1998] is used for many simulations because of the good properties such as higher dimensional uniformity [e.g., Knuth, 1981]. It is also applicable because it also needs only states. If all values of the generated random number sequence are conserved, hardware random number generators could be applicable in which physical phenomena are used. However, it is nonsense in almost all cases because very large storage is required.

If the condition is met, we can apply the new scheme for the PS. First of all, I will explain the terms like interval, segment, checkpoint and unit used in the scheme. The new scheme is explained with an example. Let us consider the case where the last time step of the time series is $T = 80$, *i.e.*, length including initial time step is $81 = 3^4$. We divide the whole time interval $[0, 80]$ into three intervals [0,26], [27,53], [54,80] and then further divide the divided intervals again into three intervals:

$$
\begin{aligned}
[0,26] &\Longrightarrow [0,8],\ [9,17],\ [18,26],\\
[27,53] &\Longrightarrow [27,35],\ [36,44],\ [45,53],\\
[54,80] &\Longrightarrow [54,62],\ [63,71],\ [72,80].
\end{aligned}
$$

These intervals are naturally associated with a two-level triplet tree shown in Figure 10.1. (The edges of the triplet tree were not drawn to avoid that the figure becomes tedious. The nine nodes at the bottom of the figure is not counted as a node of the triplet tree.) The numbers attached to the nodes (inside circles) are time points and will be taken as checkpoints in the following procedure. All

of them are not simultaneously taken as checkpoints but a part of them are assigned to checkpoints dynamically as the procedure proceeds. (There are three time points in the figure which appears in several nodes in Figure 10.1, *i.e.*, $t = 0$ (three times), $t = 27$ and $t = 54$ (two times). All the nodes with the same time point appear in Figure 10.1 are interpreted as a unique checkpoint, that is, a checkpoint is identified by a time point.) A segment is defined as the set of consecutive snapshots between the closest two checkpoints. For example, the snapshots from (time) 0 to 8, from 9 to 17, are segments. The size of a segment is nine in this example.

In order to smooth the time-series with the length 80, we need to provide $5(= 3 * 2 - 1)$ units storage u_0, u_1, u_2, u_3, u_4 to save snapshots at checkpoints and $8(= 81/(3^2) - 1)$ units storage *seg* to store a segment, namely. The alphabets and the arrows in Figure 10.1 represent the calculation order in the recursive recomputation scheme. Roughly speaking, computation is executed in the order of the alphabets A∼R attached to the nodes and edges of the graph.

The procedure is as follows. At first, prediction and filtering steps are conducted to generate $\{\boldsymbol{x}_{t|t}\}$, $(t = 1, \cdots, 80)$, with the snapshots at $i = 0$(initial particles), 27, 54, 63, 72 (A∼E in Figure 10.1; in the following, I indicate the corresponding places in the figure in the similar manner). They are stored at $u_0, \cdots, u_4$, respectively. The filtering particles $\{\boldsymbol{x}_{t|t}\}$ at $t = 73, \cdots, 80$ are also stored in *seg* (F in Figure 10.1). Then we compute smoothing distributions $\{\boldsymbol{x}_{t|80}\}$, $(t = 73, \cdots, 80)$ based on $\{\boldsymbol{x}_{t|t}\}$ stored in *seg* and $\{\boldsymbol{x}_{72|72}\}$ stored as u_4 (F). To compute the smoothing distributions at $t = 63, \cdots, 71$, the filtering distributions at $t = 64, \cdots, 71$ are calculated by the recomputation from the snapshot $u_3 = \{\boldsymbol{x}_{63|63}\}$ and are stored into the segment *seg* (G). Then we compute smoothing distributions at $t = 63, \cdots, 71$ based on *seg* and u_3 (G). In the similar manner, the smoothing distributions at $t = 54, \cdots, 62$ are obtained by recomputation and storage of the filtering distributions at $t = 55, \cdots, 62$ from $u_2 = \{\boldsymbol{x}_{54|54}\}$ (H). This finishes computation of the smoothing distributions at $t = 54, \cdots, 80$.

In Next, I explain how to compute the smoothing distributions at $t = 27, \cdots, 53$. Recall that the snapshot at the checkpoint $t = 27$ (B) were stored in u_1. We recompute filtering distributions

$\boldsymbol{x}_{t|t}$ for $t = 27, \cdots, 45$ based on u_1, with storing only at the checkpoints $t = 36$ and $t = 45$ as u_3 and u_4, respectively (I and J). Then we compute the filtering distributions at $t = 46, \cdots, 53$ based on $u_4 = \{\boldsymbol{x}_{45|45}\}$ and store all of them in the segment *seg* (K). Based on *seg* and u_4, we can obtain the smoothing distributions at $t = 45, \cdots, 53$ (K). Smoothing distributions at $t = 36, \cdots, 44$ and $t = 27, \cdots, 35$ are computed exactly in the same manner (L and M). Important thing in these steps (I~M) is that u_3 and u_4 are used to store snapshots again.

Finally, we compute smoothing distributions at $t = 0, \cdots, 26$ exactly in the same manner with starting from the snapshot $u_0 = \{\boldsymbol{x}_{0|0}\}$ instead of u_1 (N~R).

As is explained above, we reuse the storages u_3 and u_4 for computing the smoothing distributions in the interval $t = 54, \cdots, 80$ when we compute $t = 27, \cdots, 53$ and $t = 0, \cdots, 26$. For this reason, we just need $u_0, \ldots, u_4$ and *seg*. This is why the new scheme can reduce required memory. As to time complexity, we need recomputation of all filtering distributions for each level of tree after all. Therefore, we need three times as many predictions and filtering steps as in ordinary fixed-interval smoothing.

Now I discuss what is going to happen when the length of time series gets three times larger, *i.e.*, $T = 242 = 3^5 - 1$. Suppose that we store three snapshots $u_{-2} = \{\boldsymbol{x}_{0|0}\}$ and $u_{-1} = \{\boldsymbol{x}_{81|81}\}$, $u_0 = \{\boldsymbol{x}_{162|162}\}$. Then starting from u_0 we will be able to compute $\{\boldsymbol{x}_{t|242}\}$, $(t = 162, \cdots, 242)$ according to exactly the same procedure as explained above. It would be possible to execute this task with the 13 units. In the similar manner, we can compute $\{\boldsymbol{x}_{t|242}\}$, $(t = 0, \cdots, 161)$ based on u_{-1} and $u{-}2$. Therefore, even when the interval gets three times longer, yet smoothing the whole interval can be done just with two additional snapshots. As to the time complexity, we need four times as many prediction and filtering steps as in ordinary smoothing. Thus, I observed that

1. when length of time series is 81, we need five snapshots for checkpoints and eight snapshots for a segment (13 units in total), and three whole filtering steps are necessary for smoothing,

2. when length of time series is 243, we need seven snapshots for checkpoints and eight snapshots

for one segment (15 units in total) and four whole filtering steps are necessary for smoothing.

More generic form of recursive recomputation scheme is as follows. The core idea of recomputation to save storage is to store snapshots at several checkpoints and when we need a filtering distribution in smoothing, then we recover the filtering distribution by recomputation from the snapshot of the closest preceding checkpoint. In the prototype recomputation scheme, the set of checkpoints was determined in advance and remain unchanged during computation. Generally, the set of checkpoints may change dynamically as computation proceeds. In the following, I denote by Ω_t as the set of checkpoints at the time point t and denote by $snapshot(\Omega_t)$ the set of the snapshots corresponding to the checkpoints in Ω_t. Then the generic recursive recomputation scheme is written as follows.

1. Provide two areas $CHECKPOINTS$ and $SNAPSHOTS$ to store checkpoints and snapshots.
2. Compute the set of checkpoints Ω_T and let $CHECKPOINTS := \Omega_T$; Then execute the (whole) forward sweep once storing the snapshots for Ω_T at $SNAPSHOTS$. Let $t := T$.
3. While $t \neq 1$ do the steps 4, 5, 6, 7 and 8. (In the loop below, $\{\boldsymbol{x}_{t-1|t-1}\}$ and $\{\boldsymbol{x}_{t-2|T}\}$ are computed. In the beginning of this loop, we have $CHECKPOINTS = \Omega_t$ and $SNAPSHOTS = snapshot(\Omega_t)$.)
4. Let s be the checkpoint in Ω_t closest to $t-1$ and $s \leq t-1$.
5. Compute the set Ω_{t-1} of checkpoints associated with $t-1$.
6. Execute a partial forward sweep from s to $t-1$ to compute $\{\boldsymbol{x}_{t-1|t-1}\}$, constructing $snapshot(\Omega_{t-1})$. More precisely, we first remove $snapshot(\Omega_t - (\Omega_{t-1} \cap \Omega_t))$ from $SNAPSHOTS$, and then execute filtering steps from s to $t-1$ to compute $\{\boldsymbol{x}_{t-1|t-1}\}$ from $\{\boldsymbol{x}_{s|s}\}$, add $snapshot(\Omega_{t-1} - (\Omega_t \cap \Omega_{t-1}))$ to $SNAPSHOTS$.
7. Let $CHECKPOINTS := \Omega_{t-1}$ and compute smoothing particles $\{\boldsymbol{x}_{t-2|T}\}$ from $\{\boldsymbol{x}_{t-1|t-1}\}$.
8. $t := t-1$.

10.1.2 Space Complexity

I analyze the space complexity of the recursive recomputation scheme in this subsection. If the length of the time series is T and the number of stack at each level is S, the height L of the tree is bounded by $\lceil \log T / \log S \rceil$, regardless to the length of segment K. From the construction of the storage above, the number of the storage is bounded by $[(S-1)\times L+K]$ where K is the length of segment. (13(= 2×2+9) in the example). Therefore, the total storage is bounded by $M\left((S-1)\left\lceil \frac{\log T}{\log S} \right\rceil + K\right)$, where M is the unit, the storage necessary for each time point. The result shows the space complexity of the new scheme is reduced from $O(MT)$ to $O(M \log T)$ maximally, whereas the time complexity is increased.

10.2 Numerical Experiment

I implemented the PS based on the recursive recomputation scheme, and compared it with an ordinary implementation. Used PS procedure is "smoothing by storing the past state vectors [Kitagawa, 1996]." The reason why this procedure is employed is that the computational cost in the smoothing procedure can be reduced to $O(N)$ with a little additional memory whereas computational cost in other methods needs $O(N^2)$. Nikkei 225 index (Japanese stock price index) is used as the benchmark data. Nikkei 225 index was also analyzed in Kitagawa [1996] with the following model which is sufficiently low in computational cost as benchmark. The period of time is from January 1987 to August 1990, and the length of the time series is 956. The original time series is depicted in Figure 10.2. Both ordinary and new methods were applied to smooth the data with the model:

$$\alpha_t = 2\alpha_{t-1} - \alpha_{t-2} + \varepsilon_t, \tag{10.1}$$

$$p_t = 2p_{t-1} - p_{t-2} + \delta_t, \tag{10.2}$$

$$y_t = \alpha_t + w_t, \tag{10.3}$$

$$\varepsilon_t \sim \text{Cauchy}(0, \tau_1^2),\ \delta_t \sim N(0, \tau_2^2),\ w_t \sim N(0, \exp(p_t)), \tag{10.4}$$

with $\tau_1^2 = 121$ and $\tau_2^2 = 0.005$. These can be formulated as nonlinear non-Gaussian SSM (equation (2.2) and (2.3)) and the dimension of the state vector is four.

First, I show filtered trend and residual in Figure 10.3, which is obtained by the PF with $3,000,000$ particles. The median is used as estimation. It is seen that the residual gets larger around $t = 220, 830$ and 940. Main interest in this model is estimation of volatility series $\exp(p_t)$ rather than trend. In the following, I focus on volatility and demonstrate advantage of the new scheme.

The median of the filtering distribution is shown in Figure 10.4. Figure 10.5 shows the smoothed results of the ordinary implementation which uses about 5GB. Figure 10.5 is the result of smoothing the whole interval with 150,000 particles, the largest number of particles implementable with 5.6GB. Figure 10.6 shows the result of the fixed-lag smoother with $3,000,000$ particles and 40 lag-length, which is the lag available in about 5GB. The storage used in them also corresponds to the total number of the units used in the fixed interval PS with the recursive recomputation scheme.

It is seen that the median of the filtering distributions in Figure 10.4 is quite shaky. The median curve of smoothing the whole interval with 150,000 particles shown as the solid line in Figure 10.5 is much smoother than the filtering. However, the broken/dotted/chain curves in Figure 10.5 showing 2.3%, 15.9%, 84.1%, 97.7% sample points of the smoothing distributions still behave quite irregularly. This behavior derives from degeneracy, *i.e.*, the samples concentrate on few support points. Therefore, it would be difficult to infer something from obtained smoothed distributions reliably. On the other hand, the curves obtained by the fixed-lag smoother in Figure 10.6 seem to change quite smoothly. Therefore, one may think that reasonable estimates of smoothing distributions are obtained here. However, we should not simply trust this result, since the lag may not be long enough.

An approach to answer the question is to smooth the whole interval with much more particles, *e.g.* $3,000,000$ particles. The problem is that it requires 100–120GB storage to implement it with an ordinary implementation and hence would be hard. Therefore, I apply new scheme. The PS with $3,000,000$ particles and $1,000,000$ particles were implemented with the recursive recomputation

scheme and were applied to smooth the whole interval. The level L is set to three, the number of stack at each level S is also three and the length of segment K is 35 ($\approx ((T+1)/S^L)$). The total number of stored units is about 41 ($= (S-1) \times L + K$). This means that only 41 units are required to smooth whole time interval, whereas the basic approach needs 956 units. It result in reducing about 20 times from the originally required memory.

Figure 10.2 depict the results of smoothing the whole interval by the recursive recomputation scheme with $1,000,000$ and $3,000,000$ particles. Required memory is 1.7GB for 1,000,000 particles and 5.2GB for 3,000,000 particles. Used memory in the latter is as much as in the settings of Figure 10.5. However, the curves change smoothly enough compared with Figure 10.5 for a long period of $t \geq 200$.

It should be noted that the reason why the error band, the band between 2.3% and 97.7%, around the period of $t \geq 900$ is wider than other periods comes from not the degeneracy problem but the lack of future information. In other words, degeneracy problem only affects preciseness of Monte Carlo approximation in past period and does not affect smoothing distribution itself which is only determined from model and data. Therefore, increasing the number of particles can give smoothness of estimation in the past periods but can not reduce (or)the band around the period of $t \geq 900$.

The change of the number of the particles representing the smoothing distributions in the three smoothing cases are shown in Figure 10.8. The number of the particles for the truncated time steps ($t \geq 920$) are shown in Table 10.1. It is seen that the number of different particles reduces exponentially as time step t decreases. As is seen from Figure 10.8 and Table 10.1, the number of particles reduces to 50 already at $t = 780$ in the smoothing with $150,000$ and the smoothing distributions are represented about just 10 particles for $t \leq 500$. This means that it is quite difficult to rely on the smoothing results with $150,000$ particles. On the other hand, both in the case of $1,000,000$ particles and $3,000,000$ particles, more than 50 particles remain even at $t = 215$, using almost the same storage compared with the ordinary implementation. It is clear that the smoothed distributions are much

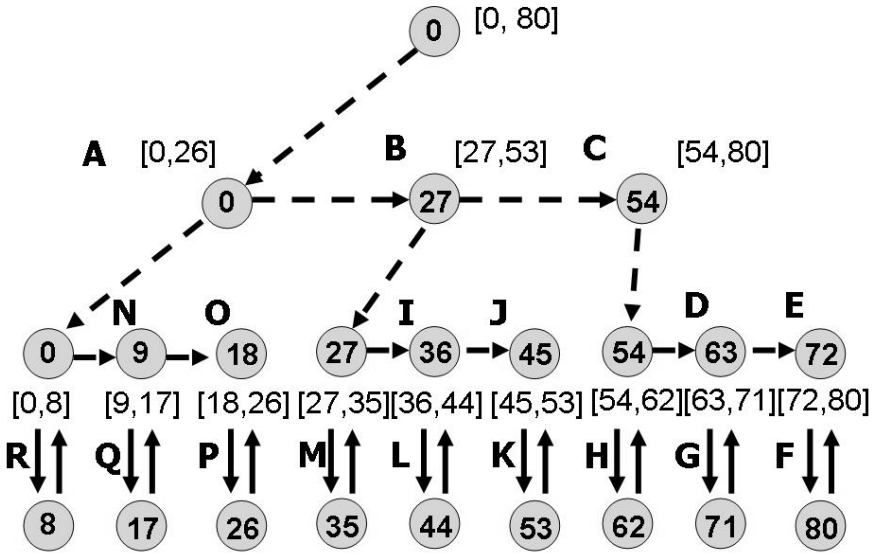

Figure 10.1: Scheme of recursive recomputation

Table 10.1: Change of the number of particles at the beginning of smoothing procedure

	$t = 956$	$t = 950$	$t = 940$	$t = 930$	$t = 920$
150,000 particles	94,681	19,870	6,128	2,860	396
1,000,000 particles	626,161	134,060	41,252	19,675	2,719
3,000,000 particles	1,885,934	403,422	122,473	58,306	7,987

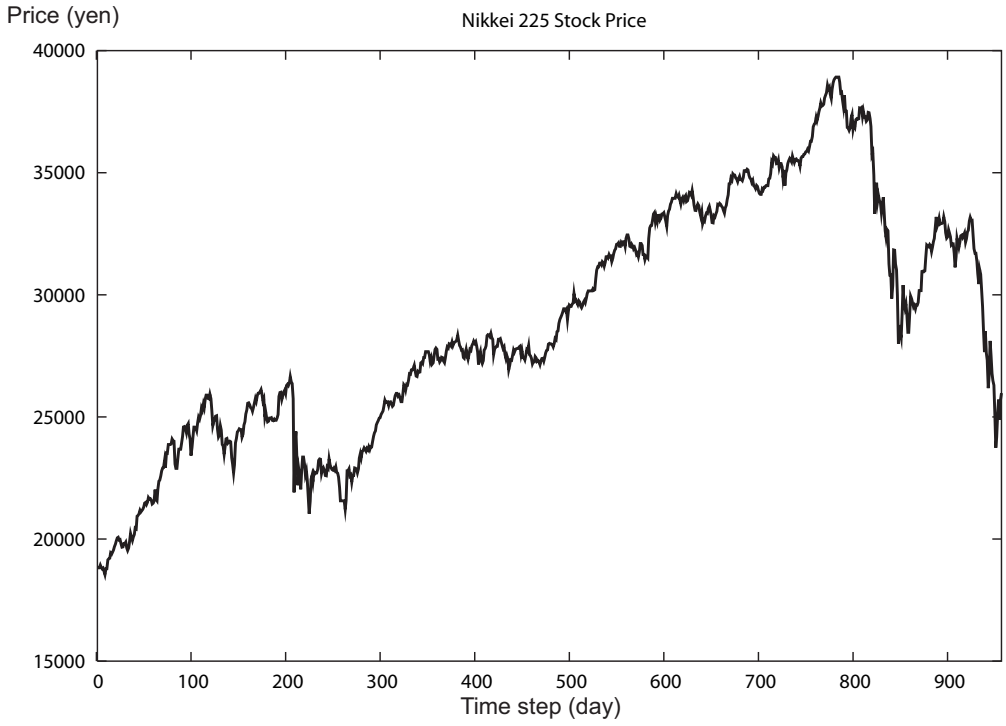

Figure 10.2: Used data (Nikkei 225 index)

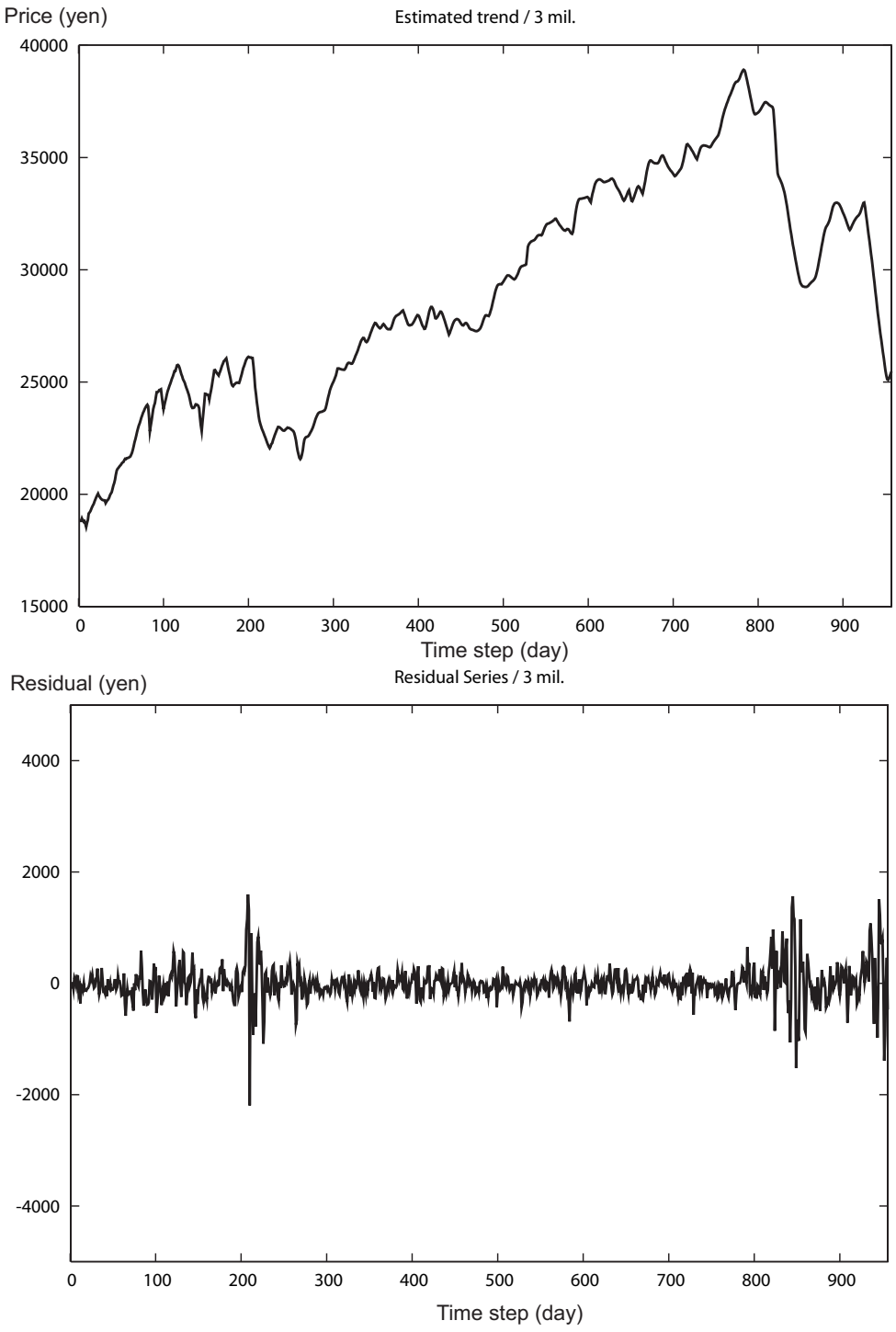

Figure 10.3: Filtered trend (above) and residual (below) series

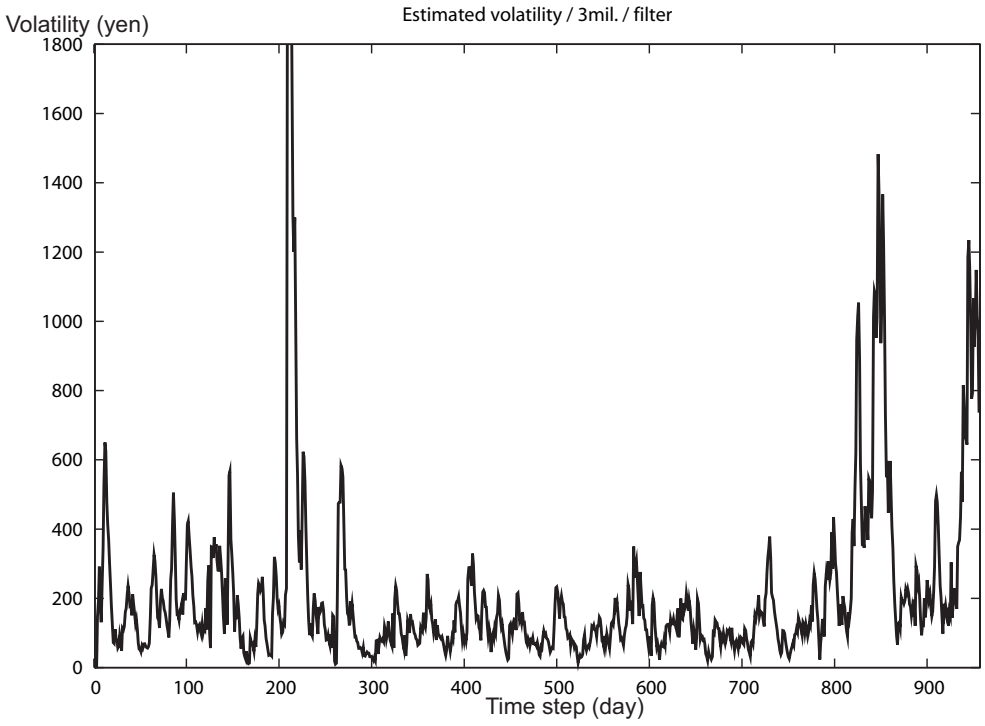

Figure 10.4: The median of the filtering distributions of volatility, 3,000,000 particles

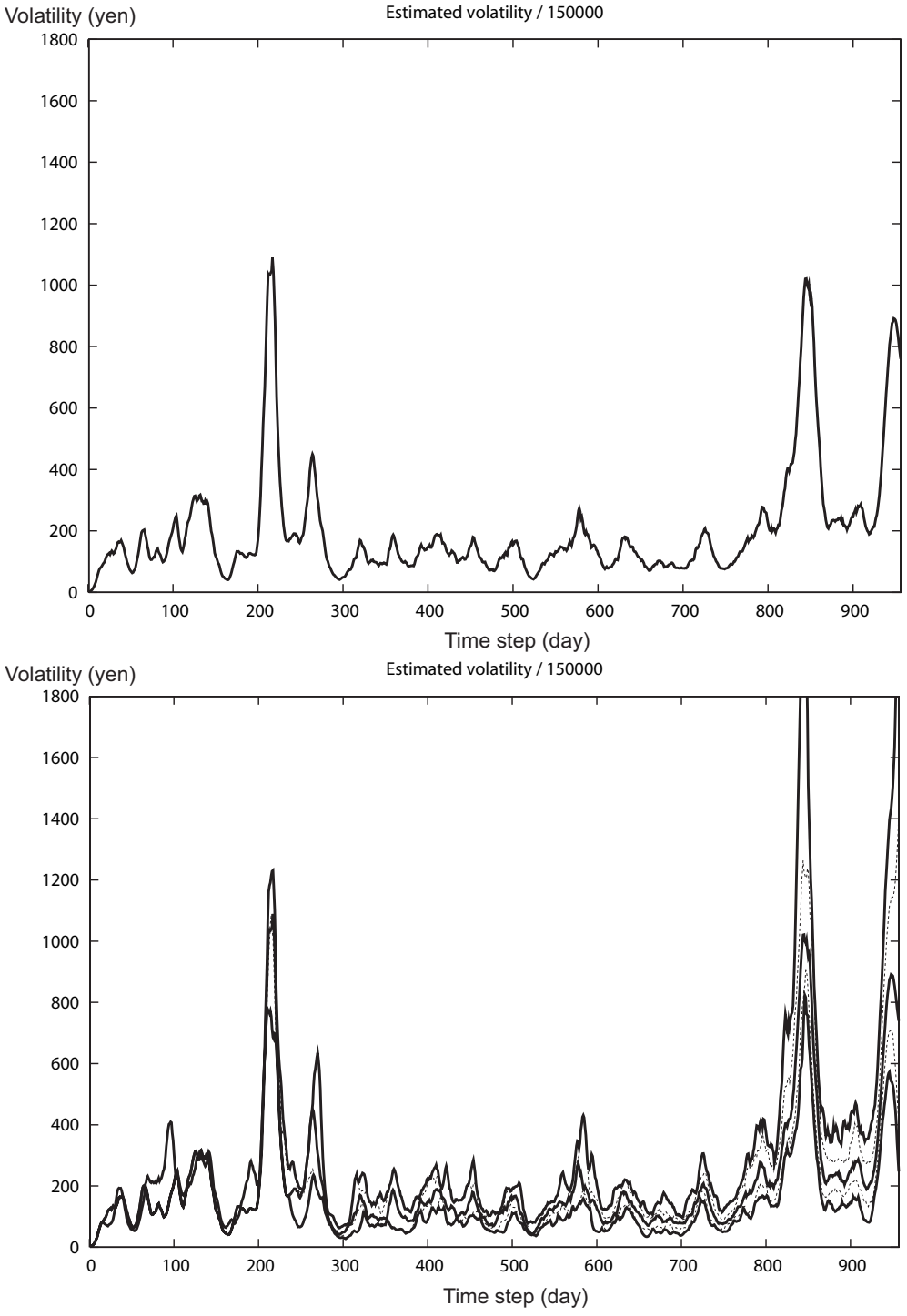

Figure 10.5: Smoothed volatility with ordinary implementation, 150,000 particles. The solid curve in the upper figure is median of the distribution. The solid curves in the lower figure are 2.3%, median and 97.7% points, which correspond to the percent point of -2σ, average and the percent point of $+2\sigma$ in normal distribution. Broken curves represent 15.9% and 84.1% points, which correspond to the percent points of $-\sigma$ and σ.

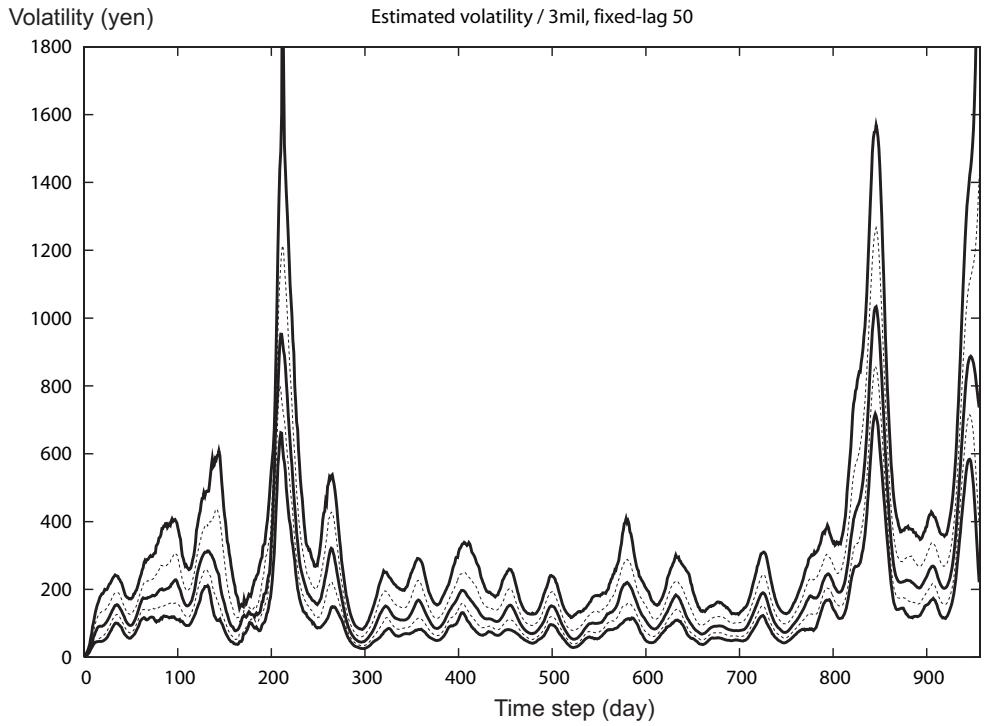

Figure 10.6: The smoothed volatility obtained by fixed-lag smoother with 3,000,000 particles. The solid curves are 2.3%, median and 97.7% points and broken curves are 15.9% and 84.1% points.

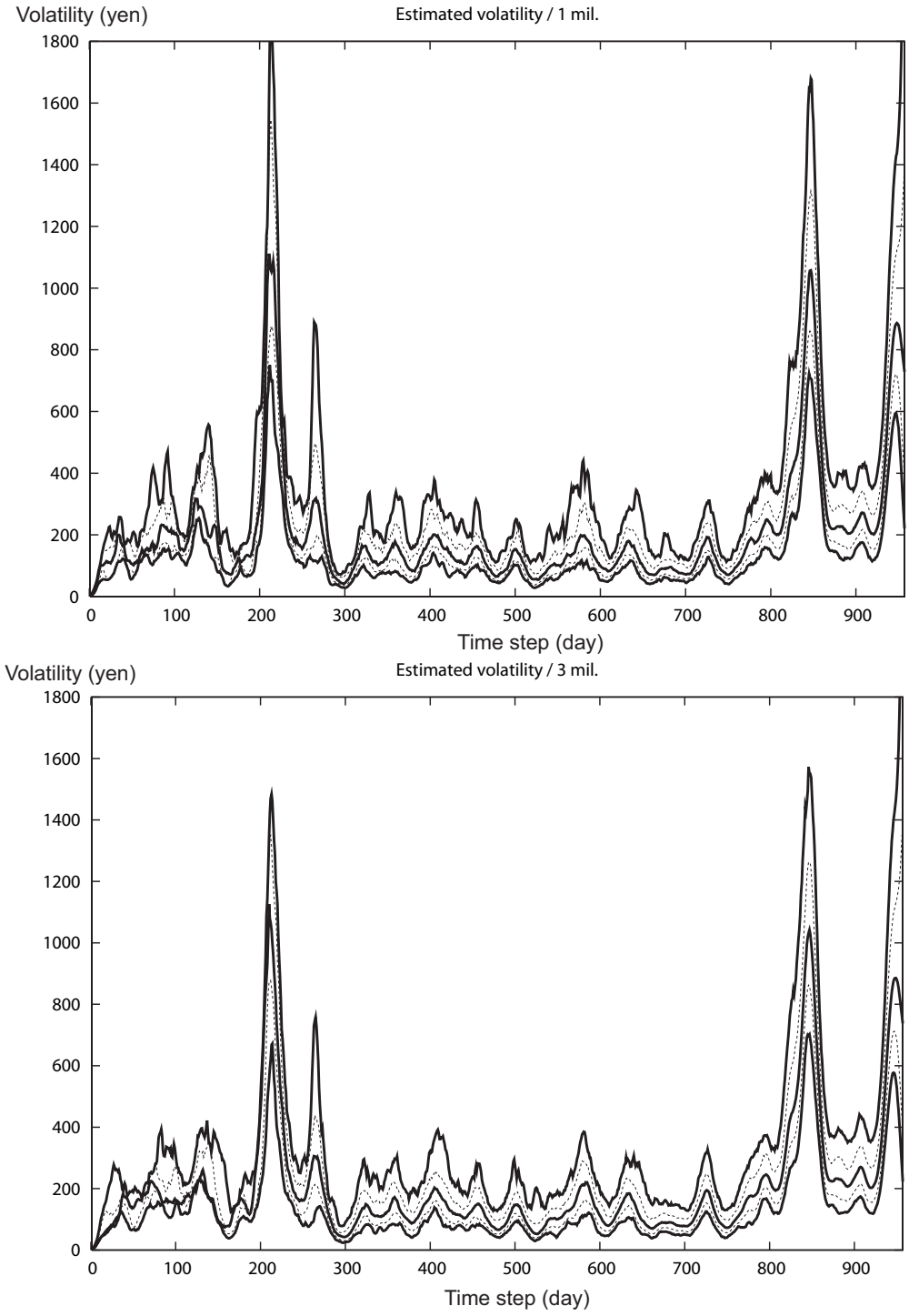

Figure 10.7: The smoothed volatility series with 1,000,000 (above) and 3,000,000 (below) particles by the recursive recomputation scheme. The solid curves are 2.3%, median and 97.7% points and broken curves are 15.9% and 84.1% points.

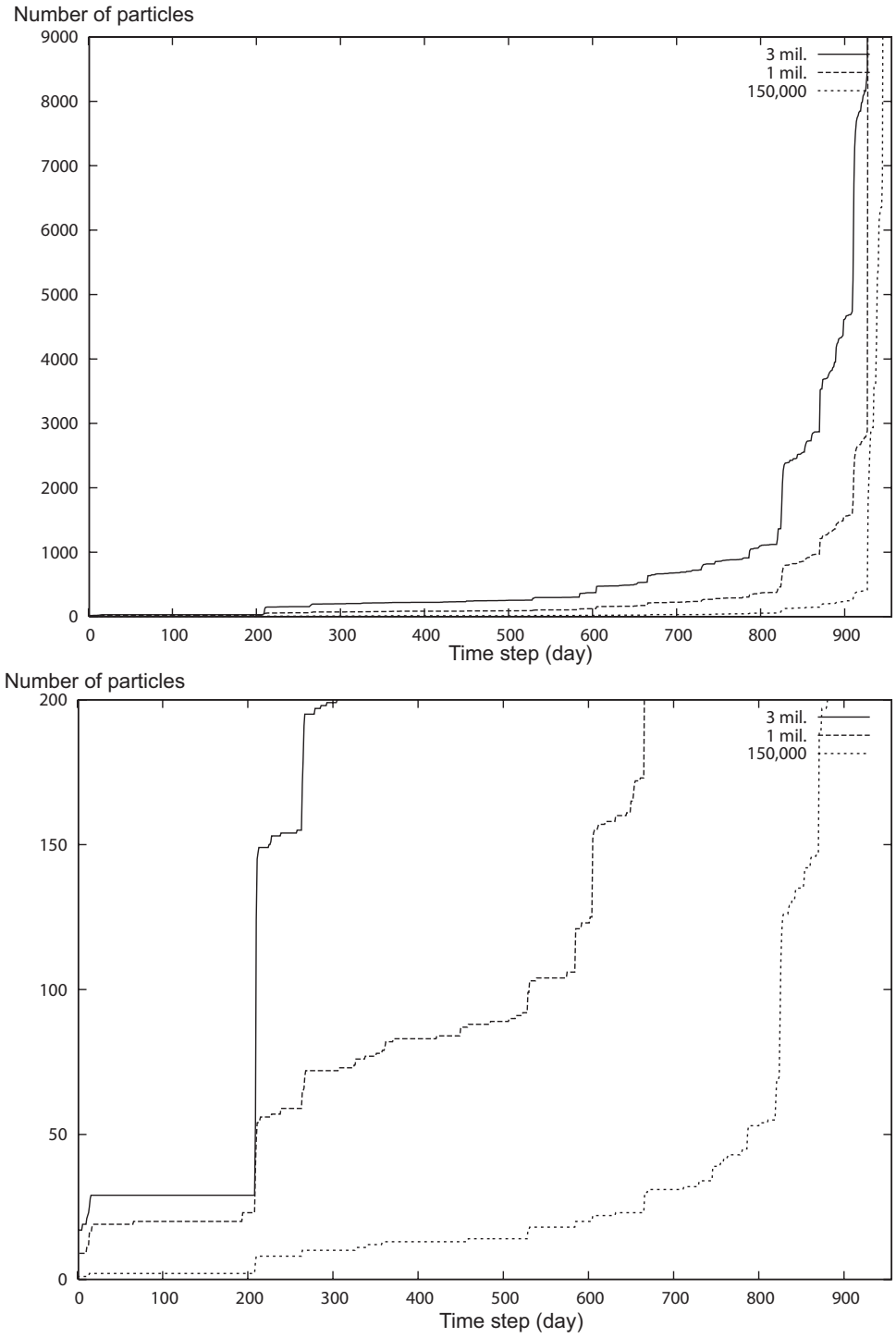

Figure 10.8: Change of the number of particles in smoothing the whole interval. Solid line: 3,000,000 particles, thick broken line: 1,000,000 particles, thin broken line: 150,000 particles; both figures plot same data in different scaling

more reliable in the latter two cases with the recursive recomputation scheme.

From the results demonstrated so far, the advantage of the recursive recomputation scheme is clear. It is seen that the fixed-lag smoothing obtained by 3,000,000 particles with the ordinary implementation and the smoothing distributions obtained by 3,000,000 particles with the recursive recomputation scheme is very similar for $t \geq 200$. From this result, it is concluded that the lag 50 is enough for the case analyzing volatility with this data.

In Table 10.2, required storage and computational time is shown. The computational experiment is executed on SGI Altix3700 SuperCluster [256CPU (Intel Itanium2 1.3GHz), 1331.2GFLOPS, 1920GB main memory], using only one CPU with 8GB memory. The following two things can be confirmed. The first point is that the required storage to implement full-interval smoothing is reduced by a factor of $1/20$ or less. This is expected results from the things because the total number of storage in the new scheme is 41 units and the length of time series is 956 and hence the ratio $(T + 1)/((S - 1) \times L + K)$ is about 23. The second point is that the elapsed time is much more required in recursive recomputation scheme than in the ordinary implementation, by a factor of 53 times. However, it is less than the 23×3 which is simple approximation of smoothing in recursive recomputation obtained from the use of memory and the number of repetition. This comes from the fact that obtaining the smoothing particles at each time step from the filtering particles does not need repetition. This is one of the advantages of the new scheme.

Table 10.2: Timing data and required storage

	Elapsed time	Storage (Kbyte)
150,000 particles (Smoothing the whole interval; Ordinary Implementation)	493s(8m13s)	5,695,664
3,000,000 particles (Fixed-lag smoother; lag 50; Ordinary Implementation)	30582s(8h29m42s)	4,941,664
1,000,000 particles (Smoothing the whole interval; Recursive Recomputation Scheme)	8411s(2h20m11s)	1,763,280
3,000,000 particles (Smoothing the whole interval; Recursive Recomputation Scheme)	25489s(7h04m49s)	5,276,880

Chapter 11

Discussion

I have conducted performance comparison among the PF, the EnKF, the PS and the EnKS through numerical experiments, using nonlinear system with nonlinear observations. The result demonstrates that the superiority of the PF(PS) to the EnKF(EnKS) in managing nonlinear observations, especially in the estimation of hyper-parameter. I have also applied the smooth PF to the same system and compared the result with that of the PF and the EnKF. The result shows that the PF and the EnKF may cause the error in the estimation of hyper-parameter compared with the smooth PF. These properties are characterized through the viewpoint of relationship to GA.

Recursive recomputation scheme have been shown as another approach to the solution against high dimensionality. The space complexity in the PS are reduced through the scheme and then the number of particles are able to be increased from the viewpoint of storage. The result of the analysis on Nikkei 225 index data represents the effectiveness of increasing particles. It is an approach to resolve the degeneracy problem.

The applicability of the smooth PF and the recursive recomputation technique to highly dimensional DA needs deeper discussions. The smooth PF needs highest computational cost among other filters I have presented because it is required that online calculation of Cholesky decomposition of sample covariance matrices. It could be critical in DA because the dimension of state vector is very large and the covariance matrix is also large and dense. Additionally, many particles would be re-

quired to give precise estimation as the result of Lorenz 96 model shows. However, some kind of localization like block diagonalization is applicable to the covariance matrices, it could be the best choice.

Obstacles in applying recursive recomputation scheme in DA are the number of parallel computations and computational time of simulation model. Recomputation is unavoidable in this scheme, increase of computational cost is also inevitable. It is also true that increase of the particle means increase of the number of simulations and hence the increase of computational cost. However, this increase of computational cost can be mitigated by the increase of the number of physical computation device or system because the forward calculation like simulation and filtering steps can be easily parallelized. The "storing the state vector" smoother used in the numerical experiment can be also parallelized easily because of requirement of $O(N)$ operation. Therefore, computational cost problem might be overcome by limiting the number of recomputation and progress of parallel computation in the future though the applicable systems are limited at present.

Acknowledgments

First of all, I would like to express my deepest gratitude to my Ph.D. course adviser Prof. Tomoyuki Higuchi in the Institute of Statistical Mathematics. He has introduced the basic things about time series analysis to me and has motivated me to work for real data analysis. He has also introduced many researchers, which is my great experience.

I would like to thank Prof. Naoki Hirose in Kyushu University. He has advised me about simulation models and other oceanographic things. He is the first and important collaborator belonging to other institute.

Prof. Takashi Tsuchiya in the Institute of the Statistical Mathematics has given many advises about my research. Chapter 11 in this thesis is collaborate work with him. I would like to express my gratitude.

I would also like to thank Prof. Byung Ho Choi in SungKyunKwan University in Korea. He introduced me many things about tsunami. My first visit of research laboratory in foreign country.

I would thank to Prof. Yosiyasu Tamura and Prof. Yoshinori Kawasaki for the advises in my composition of this doctor thesis. The advises contribute to sophisticate the thesis.

Finally, I would like to thank my parents, who have supported me and continued to encourage me to achieve my goals.

Bibliography

Anderson, J.L. (2001). An ensemble adjustment kalman filter for data assimilation. *Monthly Weather Review* **129**, 2884–2903.

Arakawa, A. and V. Lamb (1977). Computational design of the basic dynamical processes in the ucla general circulation model. In: *General circulation models of the atmosphere* (J. Chang, Ed.). Vol. 17. pp. 174–264. Academic Press.

Bennett, A. F. (2001). *Inverse Modeling of the Ocean and Atmosphere*. Cambridge University Press. Cambridge.

Bishop, C.H., B.J. Etherton and S.J. Majumdar (2001). Adaptive sampling with the emsemble transform kalman filter. part i. theoretical aspects. *Monthly Weather Review* **129**, 420–436.

Broomhead, D. S. and G. P. King (1986). Extracting qualitative dynamics from experimental data. *Physica D* **20**, 217–236.

Carlin, D. P., N. G. Polson and D. S. Stoffer (1992). A Monte Carlo approach to nonnormal and nonlinear state-space modeling. *Journal of American Statistical Association* **87**, 439–500.

Choi, B.H. and S.J. Hong (2001). Simulation of prognostic tunamis on the Korean coast. *Geophysical Research Letters* **28**, 2013–2016.

Choi, B.H., K.O. Kim and H.M. Eum (2002). Digital bathymetric and topographic data for neighbor-

ing seas of Korea. *Journal of Korean Society of Coastal Ocean Engineering* **14**, 41–50. (in Korean with English abstract).

Clapp, T.C. and S.J. Godsill (1999). Fixed-lag smoothing using sequential importance sampling. In: *Bayesian Statistics 6* (J.M. Bernardo, J.O. Berger, A.P. Dawid and A.F.M. Smith, Eds.). pp. 743–752. Oxford University Press. Oxford.

Courtier, P., T. Thepaut and A. Hollingsworth (1994*a*). A strategy for operational implementation of 4DVAR, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society* **120**, 1367–1387.

Courtier, P., Thepaut and A. Hollingsworth (1994*b*). A strategy for operational implementation of 4dvar, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society* **120**, 1367–1387.

Daley, R. (1991). *Atmospheric Data Analysis*. Cambridge University Press. Cambridge.

Doucet, A., J.F.G. de Freitas and N.J. Gordon (2001). *Sequential Monte Carlo Methods in Practice*. Springer-Verlag. New York.

Doucet, A., S.J. Godsill and C. Andrieu (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing* **10**, 197–208.

Durbin, J. and S. J. Koopman (2001). *Time Series Analysis by State Space Methods*. Oxford University Press. Oxford.

Efron, B. and R. J. Tibshirani (1993). *An Introduction to the Bootstrap*. Chapman Hall. New York.

Evensen, G. (1994). Sequential data assimilation with a non-linear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research* **99**(C5), 10 143–10 162.

Evensen, G. (2003). The ensemble Kalman filter: Theoretical formulation and practical implementation. *Ocean Dynamics* **53**, 343–367.

Evensen, G. (2004). Sampling strategies and square root analysis schemes for the enkf. *Ocean Dynamics* **54**, 539–560.

Evensen, G. and P.J. van Leeuwen (2000). An ensemble Kalman smoother for nonlinear dynamics. *Monthly Weather Review* **128**, 1852–1867.

Godsill, S.J., A. Doucet and M. West (2004). Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association* **99**, 156–168.

Gordon, N. J., D. J. Salmond and A. F. M. Smith (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F* **140**(2), 107–113.

Harvey, A.C. (1989). *Forecasting, Structural Time Series models and the Kalman Filter*. Cambridge University Press. Cambridge.

Higuchi, T. (1996). Genetic algorithm and Monte Carlo filter. *Proceedings of Institute of Statistical mathematics* **44**(1), 19–30. (in Japanese with English Abstract).

Higuchi, T. (1997). Monte Carlo filter using the genetic algorighm operators. *Journal of Statistical Computation and Simulation* **59**, 1–23.

Iba, Y. (2001). Population Monte Carlo algorithms. *Transactions of the Japan Society for Artificial Intelligence* **16**, 279–286.

Japan Meteorological Agency (1993). Past weather data at each observation point. http://www.data.kishou.go.jp/etrn/index.html, (in Japanese).

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering* **82**, 35–45.

Kitagawa, G. (1987). Non-Gaussian state-space modeling of nonstationary time series (with discussion). *Journal of the American Statistical Association* **82**, 1032–1063.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space model. *Journal of Computational and Graphical Statistics* **5**(1), 1–25.

Kitagawa, G. (1998). Self-organizing state space model. *Journal of the Americal Statistical Association* **93**, 1203–1215.

Kitagawa, G. and S. Sato (2001). Monte Carlo smoothing and self-organising state-space model. In: *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas and N. Gordon, Eds.). pp. 177–195. Springer. New York.

Kitagawa, G. and W. Gersch (1996). *Smoothness Priors Analysis of Time Series (Lecture Notes in Statistics)*. Springer-Verlag. New York.

Knuth, D.E. (1981). *The Art of Computer Programming. Vol. 2: Seminumerical Algorithms*. Addison-Wesley. Reading, Massachusetts.

Lorenz, E. N. and K. A. Emanuel (1998). Optimal sites for supplementary weather observations: Simulation with a small model. *Journal of Atmospheric Sciences* **55**, 399–414.

Majumdar, S.J., C.H. Bishop, B.J. Etherton, I. Szunyogh and Z. Toth (2001). Can an ensemble transform kalman filter predict the reduction in forecast-error variance produced by targeted observations?. *Quarterly Journal of the Royal Meteorological Society* **127**, 2803–2820.

Matsumoto, M. and T. Nishimura (1998). Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation* **8**(1), 3–31.

MIRC (2003). JTOPO1—Northwest Pacific one minute grid data. Marine Information Research Center Japan Hydrographic Association, http://www.mirc.jha.or.jp/products/JTOPO1/, CD-ROM, (in Japanese).

Nakamura, K. and T. Tsuchiya (2006). A recursive recomputation approach for smoothing in non-linear state space modeling – an attempt for reducing space complexity –. *IEEE Transaction on Signal Processing*. submitted.

Nakamura, K., G. Ueno and T. Higuchi (2005). Data assimilation : Concept and algorithm. *Proceedings of the Institute of Statistical Mathematics* **53**(2), 201–219. (in Japanese with English abstract).

NAVOEANO (n.d.). Digital bathymetric data base - variable resolution. Naval Oceanographic Office.

Ott, E., B. R. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, M. Corazza, E. Kalnay, D. J. Patil and J. A. Yorke (2004). A local ensemble Kalman filter for atmospheric data assimilation. *Tellus* **56A**, 415–428.

Pham, D.T. (2001). Stochastic methods for sequential data assimilation in strongly nonlinear systems. *Monthly Weather Review* **129**, 1194–1207.

Pham, D.T., J. Verron and M.C. Roubaud (1998). A singular evolutive extended kalman filter for data assimilation in oceanography. *Journal of Marine Systems* **16**, 323–340.

Press, W.H., B.P. Flannery, S.A. Teukolsky and W.T. Vetterling (1992). *Numerical Recipes in C*. Cambridge University Press. Cambridge.

Satake, K. (1989). Inversion of tsunami waveforms for the estimation of heterogeneous fault motion of large earthquakes: The 1968 Tokachi-oki and the 1983 Japan Sea earthquakes. *Journal of Geophysical Research* **94**, 5627–5636.

Smith, W.H.F. and D.T. Sandwell (1997). Global sea floor topography from satellite altimetry and ship depth soundings. *Science* **277**, 1956–1962.

Stavropoulos, P. and D. M. Titterington (2001). Improved particle filters and smoothing. In: *Sequential Monte Carlo Methods in Practice* (A. Doucet, N. de Freitas and N. Gordon, Eds.). pp. 295–317. Springer. New York.

Stavropoulos, P. and D.M. Titterington (1998). Computational Bayesian filtering. Technical report. Technical report 98-13, Department of Statistics, University of Glasgow.

Talagrand, O. and P. Courtier (1987). Variational assimilation of meteorological observations with the adjoint vorticity equation i: theory. *Quarterly Journal of the Royal Meteorological Society* **113**, 1311–1328.

Tanioka, Y. and K. Satake (2001). Coseismic slip distribution of the 1946 Nankai earthquake and aseismic slips caused by the earthquake. *Earth Planets Space* **53**, 235–241.

Tanioka, Y., Yudhicara, T. Kususose, S. Kathiroli, Y. Nishimura, S. Iwasaki and K. Satake (2006). Rupture process of the 2004 great Sumatra-Andaman earthquake estimated from tsunami waveforms. *Earth Planets Space* **58**, 203–209.

Titov, V.V., F.I. González, E.N. Bernard, M.C. Eble, H.O.Mofjeld, J.C. Newman and A.J. Venturato (2005). Real-time tsunami forecasting: Challenges and solutions. *Natural Hazards* **35**, 41–58.

van Leeuwen, P.J. (2001). An ensemble smoother with error estimates. *Monthly Weather Review* **129**, 709–728.

Vautard, R. and M. Ghil (1989). Singular spectrum analysis in non-linear dynamics, with applications to paleoclimatic time series. *Physica D* **35**, 395–424.

Wunsch, C. (1996). *The Ocean Circulation Inverse Problem*. Cambridge University Press. Cambridge.