國立臺灣大學電機資訊學院電機工程學系

碩士論文

Department of Electrical Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

利用結構性支撐向量機進行電腦擬人音樂演奏

Computer Expressive Music Performance Using Structural Support Vector Machine

呂　行

Shing Hermes Lyu

指導教授：鄭士康博士

Advisor: Shyh-Kang Jeng, Ph.D.

中華民國 103 年 6 月

June, 2014

國立臺灣大學
電機工程學系

碩士論文

利用結構性支撐向量機進行電腦擬人音樂演奏

呂 行 撰

103
6

國立臺灣大學（碩）博士學位論文
口試委員會審定書

論文中文題目

論文英文題目

本論文係○○○君（○學號○）在國立臺灣大學○○學系、所完成之碩（博）士學位論文，於民國○○年○○月○○日承下列考試委員審查通過及口試及格，特此證明

口試委員：

______________________________（簽名）

（指導教授）

__________________　　__________________

__________________　　__________________

__________________　　__________________

__________________　　__________________

系主任、所長 ______________________（簽名）

（是否須簽章依各院系所規定）

致謝

中文摘要

請打開並編輯abstractCH.tex

關鍵字：壹、貳、參、肆、伍、陸、柒

Abstract

Open and edit abstractEN.tex

Key words:A, B, C, D, E, F, G

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 2

Previous Works

Chapter 3

Proposed Method

3.1 Overview

The high-level archtecture of the purposed system is shown in figure 3.1. The system is has two phases, the upper half of the figure is the learning phase, while the lower half is the generating phase. In the training phase, score and expressive performance recording pairs are feed in, their features will be extracted and used as the input for the learning algorithm. The learning algorithm will produce a learned model, which can be used to generate expressive performance hereafter. In the generating phase, a score is taken as input, and it's score features are extracted. The generation module ("Applying Model" box) then use the extracted features and the learned model to produce the preformance features for the score. The performance features can be viewed as the instruction for expression. With these features, an expressive MIDI performance for the input score can easily be generated.

The system is not intend to add fixed expression to all pieces. Rather it is intended to perform music according to the style which the user wants. This kind of user interactivity can be achieved in two ways: first, the user can choose the training dataset. The dataset is organized by tags, example of tags are like performer, mood, emotion, genre, style, etc. So the user can select a subset of training sample by selecting tags. Second, the phrasing is given by the user. Since phrasing controls the overall structural interpretation of a music piece, and form a breathing feeling in music, user is given direct control over

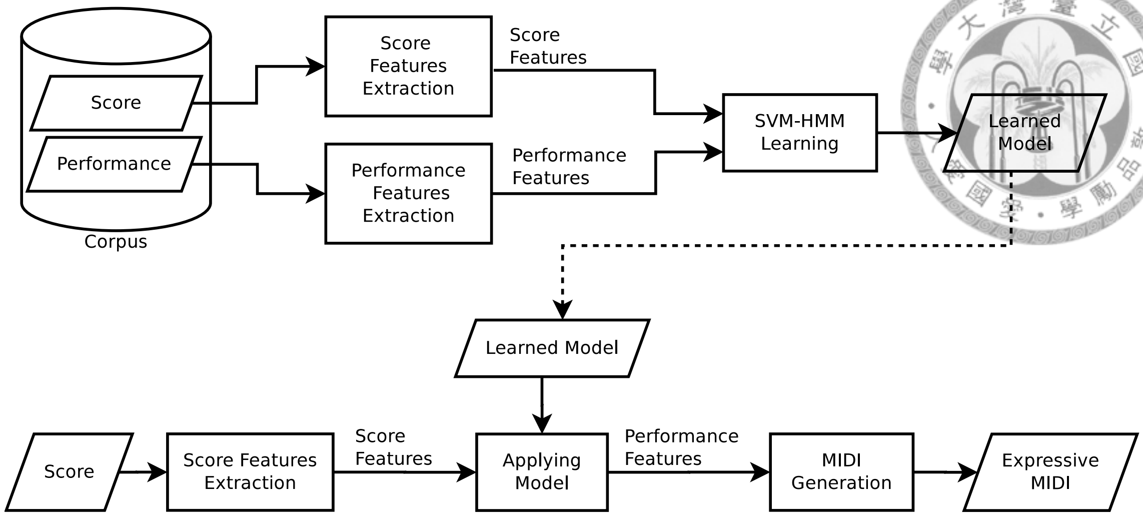

Figure 3.1: System Architecture

the performance.

There are still some constrains for the system now. The scores must be monophonic and contains only one musical phrase. One has to manully label the phrasing for any scores used. The learning algorith, namely structural suppor vector machine, can only perform offline learning, so the learning phase can only work in a non-realtime scenario. The generating phase can work much faster though, it can procude expressive music almost instantely after loading the score. All the scores are loaded in batch, the systme currently don't accept streaming input.

3.2 Features

The system is trying to mimic the process of human performance: the musican reads the explict and implict cues from the score and transform them into musical expressions. So the features can be categorized into two category: score features and performance features. Score features are information contained in the score. Performance features corresponds to the musical expression. The basic time unit for both features are a note.

3.2.1 Score Features

Score features includes:

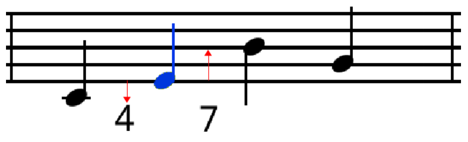

Figure 3.2: Interval from/to neighbor notes

Relative position in a phrase: The relative position of a note in the phrase. From 0% to 100%. This feature can catch the musical hint of the opening and closing of a phrase.

Relative pitch: The pitch (in semitone) of a note relative to the pitch range of the phrase. For a phrase of n notes with pitch $P_1, P_2, \ldots, P_n$,

$$RP = \frac{P_i - min(P_1, P_2, \ldots, P_n)}{max(P_1, P_2, \ldots, P_n) - min(P_1, P_2, \ldots, P_n)}$$

Where P_i is the pitch of note at position t

Interval from the previous note: The direction of melody movement. Measured in semitone.

$$IP = P_i - P_{i-1}$$

See figure 3.2 for example.

Interval to the next note: The direction of melody movement.

$$IN = P_{i+1} - P_i$$

See figure 3.2 for example.

Note duration: The duration of a note in beats.

Relative Duration with the previous note: The duration of a note divided by the duration of its previous note. For a phrase of n notes with duration $D_1, D_2, \ldots, D_n$,

$$RDP = \frac{D_i}{D_{i-1}}$$

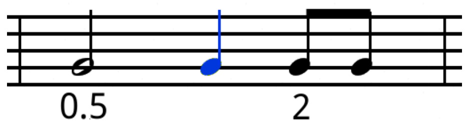

Figure 3.3: Duration from/to neighbor notes

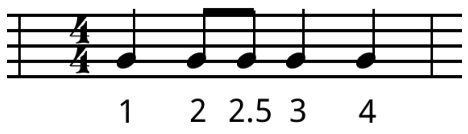

Figure 3.4: Metric position

See figure 3.3 for example.

Relative duration with the next note: The duration of a note divided by duration of its next note.

$$RDN = \frac{D_i}{D_{i+1}}$$

See figure 3.3 for example.

Metric position: The position of a note in a measure, measured by the beat unit defined by the time signature. For example, a $\frac{4}{4}$ time signature will have a beat unit of a quarter note. So if the measure consists of four quarter notes, each of them will have metric position of 1, 2, 3 and 4. See figure 3.4.

3.2.2 Performance Features

Performance features includes:

Relative onset time bias: The onset time of a recording will not be exactly as the ones indicated on the score. Given a fixed tempo (beats/second), the score timing of each note can be calculated as tempo $\times$ (beats from the start of phrase). The relative onset time bias is the difference of onset timing between the performance and the score, divided by the total length of the phrase. Namely,

$$ROB = \frac{O_i^{perf} - O_i^{score}}{length(phrase)}$$

Where O_i^{perf} is the onset time of note i in the performance, O_i^{score} is the onset time of note i in the score.

Relative loudness: The loudness of a note divided by the maximum loudness in the phrase. Measured by MIDI velocity level.

$$RL = \frac{L_i}{max(L_1, L_2, \ldots, L_n)}$$

Relative duration: The actual duration of note divided by the total length of the phrase.

$$RD = \frac{D_i^{perf}}{length(phrase)}$$

3.3 Melodic Similarity and Sample Selection

Once a score is given to the system for playing, all the samples in the database will be ranked by the melodic similarity with the given score. Here we use the melodic distance function provided by the MIDI Toolbox [**?**], which is defined as follows:

1. Melodic contour is calculated by connecting each note's pitch, forming a piece-wise linear contour.
2. Subtract the contour by it's mean to preserve only the relative part.
3. If the two phrases has different length, re-sample both phrases with fixed intervals so both of the phrase will have contour vector of the same length.
4. The L1 norm (a.k.a Taxicab distance) of these two contour vector is the similarity measure.

The reason I choose melodic contour is because it yields best results in finding melodic similarity, which is shown in [?].

3.3.1 Normalizing Onset Timing

3.4 Learning Phase

In the learning phase, the features extracted in the previous stage is feed into a machine learning algorithm to produce a phrase model. The learing module has a input/output interface that is independent of the underlying algorithm, so different algorithm can be implemented without changing the overall structure of the system.

The input of the learing module is a json file containing a list of training samples. Each sample contains the extracted score and performance features. The learning algorithm can then do any pre-processing on the features, such as aggregation or quantization. The output of this module is the algorithm specific model description. For example, a linear regression algorithm will output the regression parameters. The algorithm is requried to produce a model file containing the model description, but the systme doesn't care about the internal format of the model description file, it will simply feed this model file to the generation module in the generation stage. So the developer of the learing module has to implemnt methods to write and read the model file themselves.

In the early stage of this research, linear regression is used. The results of linear regression is shown in [?]. In this thesis, Structural Support Vector Machine [?] is the algorithm of choice. The detail of Structural SVM will be in the next Chapter.

3.5 Generating Phase

After the model for a input phrase is generated, we can than use the score features and model coefficients to calculated the performance parameters. These performance parameters will then be applied to the input score.

Some post-processing will be made for each performance parameters: The first time bias will be reduced if it is too negative and create a negative onset time for the first note. Loudness will be shift and shrink to a predefined range. The default is 80 127 MIDI loudness level. This will ensure the loudness in the output will be in acceptable range. So after we input an input score, its score features will be extracted. These score features combined with regression model will be used to calculate performance features. The result will be an expressive MIDI output. Ready to be played by hardware or software synthesizer.

Chapter 4

Structural Support Vector Machine

Chapter 5

Corpus

Since this research is based on a learning algorithm, a good expressive performance corpus is essential to its success. In this chapter, we will discuss what makes a good corpus and how to produce it.

A expressive performance corpus is a set of performance samples. Each sample consists of a score and a recording. The score is simply the music score being played, which contains notational information; and the recording is a digital or audio recording of a human musician playing the said score. With the two elements, a learning algorithm can learn how the music notation is translated into real performance. These two elements can come in many format, in the next sections we will discuss the pros and cons of some possible candidate.

5.1 Score

There are many way to represent a music score in machine-readable format, such as MusicXML [?], LilyPond [?], Finale, Sibelius, ABC, MuseData, and Humdrum. For more information, please check out [?]. For research purpose, proprietary format like Finale and Sibelius is abandoned because of their limited support from open source tools. MusicXML is based on XML (eXtensible Markup Language), it can express most music notations and metadata. LilyPond is a LaTeX-like language for music typesetting. ABC, MuseData and Humdrum are based on ASCII codes and each defines their unique representation for music score. Other formats such as image files (scanned or typesetted by computer) or PDF files are an alternative, but they are not an option for direct computer analysis. MIDI can also be shown as music score in some music notation software, but MIDI is design as control signals for digital music equipments, so it lacks some music notations. Furthermore, the model of low-level music instrument control signals doesn't fit well with music notation, so it is not considered a good way to representation for music score.

In this research, we use MusicXML as the main vehicle for music score, because of the following reasons: first, it covers most music notations and metadata need for this research. Second, it is supported in most music notation software, including the one used in this research -- MuseScore. Finally, the music21 toolbox can convert many other formats into MusicXML without problem.

5.2 Recording

A recording of expressive performance can be in MIDI or audio such as PCM WAV. Although audio recording has higher fidelity than MIDI, it takes extra effort to be analyzed by computer. Since onset detection and pitch detection algorithms still can't achieve perfect accuracy, manually labeling each notes required, which will soon become impossible to do when the texture of music become polyphonic. On the other hand, a multichannel MIDI recording can provide exact timing, key pressure, and pitch for each note, but it lacks timber and envelope information, which makes it hard to analyze how the musician use instrument-specific skills. Considering the above arguments, MIDI is used in this research.

A human musician can't play every note exactly on the beat, even if playing along with a metronome. There are two ways to record this behavior: first, record the exact note-on and note-off time, while keeping the tempo fixed; second, keep the notes on the beat, so the MIDI looks the same as the score. Then insert tempo-change event between each notes, so notes of the same length can be rendered differently because they have different tempo marker. The second method may look smart, because the score and performance can be stored in one MIDI file instead of two, but it would involve complex calculation when linearly scaling the tempo. Since tempos from different samples need to be normalized during feature extraction, the first method is superior than the second.

5.3 Existing Corpora

5.4 Corpus Used

There are a few assumptions for the corpus used:

1. All the samples are monophonic. They only contain single line of melody, without chords.
2. A song is divided into phrases by manually labeling.
3. No human error exist in the recording; the score and recording are matched note-by-note.
4. The tempo label in MIDI recordings are the tempo by which the musician played.

For our corpus, we choose Clementi's Sonatina Op. 36. This is because Clementi's Sonatina is a basic repertoire almost every piano student in Asia will learn, so it' s easy to find performers with different skill level to record the corpus. Clementi wrote these sonatinas in classical style, so the skill required to play them can be easily extended to other classical era works like Mozart or Haydn. This fact makes the learned model a very general one, which is good for evaluation. The digital score used is downloaded from KernScore website [?]. The original format is in Hundrum file format (.krn), then transformed into MusicXML by music21 toolkit. Because this research focus on monophonic melody only, so the chords in the scores are manually reduced to the highest note in the chord, which is usually the most salient melody line. All the other possible errors in the downloaded score is manually corrected to make them ready for printing and computer analysis. We use MuseScore notation editor to view and edit MusicXML; some metadata errors are corrected by editing the MusicXML with text editors .

To record the MIDI performances, we used a Yamaha MIDI keyboard with pressure sensitive keys. The Yamaha keyboard was connected to a MIDI-to-USB converter so it acted as a USB MIDI device on our Linux computer. On the Linux computer the Rosegarden Digital Audio Workstation (DAW) is used to record the MIDI. The Rosegarden DAW also generated the metronome sound to help the performer maintain a steady speed. One may argue that the tempo variation is also a part of the expression, but if the performer plays freely, the tempo is hard to determine afterwards, which will make learning scaling the performance in time domain very hard. The performers were allowed to record multiple times and edit the recording until there were no mistakes. The human error model is not part of this research now, so no mistakes are allowed in the performances to keep the problem simple.

After the MIDIs are recorded, some utility scripts we wrote are employed to make sure each recording is matched note-to-note with the corresponding score; if not, we will manually correct those mistakes. For example, if the pitch was played wrongly, we will correct the pitch but keep the onset, duration and intensity as is. The matched score and MIDI pair are then splat into phrases according to a phrasing file. Each line in the phrasing file is the starting point of each phrase. The starting point is defined as the onset timing (in quarter notes) of the first note in a phrase in the score. The phrasing is assigned by the researcher now for accuracy, but any phrasing algorithm can be applied.

Because of the scope of the research, we didn't use all the information available, here are a few pieces of potentially useful information:

1. Polyphonic recording can provide information for polyphonic performance
2. The gaps between phrases can be learned to create models for inter-phrase delay.
3. Recordings with human error can make the system play more like human.

Chapter 6

Experiments and Results

Chapter 7

Conclusions

Appendix A

Software Tools for Music Research

Appendix B

Guide

這裡將簡單介紹如何利用 LATEX 來編輯你的畢業論文，若不知道 LATEX 是什麼或是沒有概念的話，建議你可以簡單看過放在此資料夾裡的李果正 -大家來學 LATEX前四章內容，在下載適合的 LATEX 整合發行套件之後（請看第 III.項），可以嘗試用剛安裝好的 LATEX 編輯器來編譯thesis.tex這份文件，編譯的方法可以看下面第 V.項的介紹，若編譯成功，所編譯出來的 thesis.pdf 文件的應該會跟此 demo.pdf 文件一模一樣，而且沒有任何問號符號，走到這一步的話，就差不多可以開始邊學習 LATEX 邊編輯你的畢業論文了！基本上會使用到的指令都包含在論文的的各章節裡，怎麼在論文裡寫公式或是放圖之類的就自行看 tex 檔學吧。如果有任何問題或建議可以來信與我討論，我的信箱是dran31545@gmail.com，或是到此範本Google Project裡面的Issues貼上你的問題與建議，我會盡我所能更新此範本，也歡迎大家自行重製、改良此範本並散布給他人，祝大家順利畢業！

要編輯致謝請打開acknowledgementsCH.tex

I. 此範本參考並修改自下列網站的資料：

- 如何用 LaTeX 排版臺灣大學碩士論文

  —台灣大學論文 LATEX 樣版原創者黃子桓的教學網頁

- LaTeX 常用語法及論文範本

  —Hitripod所修改的範本，這裡參考了許多他所寫的格式和內容

- 使用 LaTeX 做出精美的論文
- XeTeX：解決 LaTeX 惱人的中文字型問題
- 台灣大學碩士、博士論文的 Latex 模板

II. 幾個有用的參考資料及網路資源：

- 李果正 -大家來學 LaTeX—建議先看完前四章
- WIKIBOOKS-LaTeX—好用的線上工具書
- Working with a .bib file using JabRef
- Using BibDesk - A short tutorial
- LaTeX for Physicists

III. 下載 LaTeX 整合發行套件，可參考TeX Collection：

1. MacTeX: For **MacOSX**，下載MacTeX.pkg
2. ProTeXt: For **Windows**，下載ISO file
3. TeX Live: For **GNU/Linux** and **MacOSX**, and **Windows**，下載ISO file
4. CTAN: The Comprehensive TeX Archive Network.

IV. 好用的程式：

- 文獻管理系統：
  1. JabRef

     可參考Working with a .bib file using JabRef或是Google及YouTube
  2. BibDesk (For Mac)

     可參考Using BibDesk - A short tutorial或是Google及YouTube
- 方程式編輯器：Daum Equation Editor (Chrome App，必須使用 Google 瀏覽器)

V. 編譯流程：

1. `xelatex thesis`

   對 thesis.tex 進行第一次 XeLaTeX 編譯，產生 thesis.pdf 以其他檔案

2. `bibtex thesis`

   對 thesis.tex 進行 BibTeX 編譯，產生 bbl 檔以及 blg 檔

3. `xelatex thesis`

   對 thesis.tex 進行第二次 XeLaTeX 編譯，產生目錄、圖表連結及參考文獻

4. `xelatex thesis`

   對 thesis.tex 進行第三次 XeLaTeX 編譯，產生參考文獻連結，完成編譯

注意！此範本使用 cite 套件，可依據你利用文獻管理系統所整理好的thesisbib.bib檔在論文最後產生參考文獻頁面，若你的系所規定要在每個章節的後面產生參考文獻，則可以用 chapterbib 套件，來對每個有附參考文獻的章節 tex 檔進行一次 BibTeX 編譯產生 bbl 檔，如範例的introduction.tex、THM.tex和EXP.tex，如果有這需要請把thesis.tex檔裡使用 cite 套件的指令利用註解符號% 來取消使用 cite 套件，並刪去出現在使用 chapterbib 套件指令前面的註解符號% 來啟動使用 chapterbib 套件

```
\usepackage{cite}
%\usepackage{chapterbib}
```

改 成

```
%\usepackage{cite}
\usepackage{chapterbib}
```

再來利用註解符號% 取消會把參考文獻放在論文最後的指令

```
\bibliographystyle{unsrt}
\addcontentsline{toc}{chapter}{\bibname}
\bibliography{thesisbib}
```

改 成

```
%\bibliographystyle{unsrt}
```

```
%\addcontentsline{toc}{chapter}{\bibname}
%\bibliography{thesisbib}
```

再把用來輸入章節檔案的 `\input` 指令改成 `\include` 指令

```
\input{introduction}  =>  \include{introduction}
\input{THM}           =>  \include{THM}
\input{EXP}           =>  \include{EXP}
```

最後記得在每個有附參考文獻的章節加上產生參考文獻的指令，即在introduction.tex、THM.tex和EXP.tex三個檔案裡最後啟動下面兩行指令

```
%\bibliographystyle{unsrt} => \bibliographystyle{unsrt}
%\bibliography{thesisbib}  =>  \bibliography{thesisbib}
```

而編譯時則需要對有附參考文獻的introduction.tex、THM.tex和EXP.tex各做一次BibTeX 編譯，編譯流程如下

1. `xelatex thesis`

   對 thesis.tex 進行第一次 XeLaTeX 編譯，產生 thesis.pdf 及其他檔案

2. `bibtex introduction`

   對 introduction.tex 進行 BibTeX 編譯，產生 bbl 檔以及 blg 檔

3. `bibtex THM`

   對 THM.tex 進行 BibTeX 編譯，產生 bbl 檔以及 blg 檔

4. `bibtex EXP`

   對 EXP.tex 進行 BibTeX 編譯，產生 bbl 檔以及 blg 檔

5. `xelatex thesis`

   對 thesis.tex 進行第二次 XeLaTeX 編譯，產生目錄、圖表連結及參考文獻

6. `xelatex thesis`

   對 thesis.tex 進行第三次 XeLaTeX 編譯，產生參考文獻連結，完成編譯

VI. 補充說明與注意事項：

- 口試委員會審定書：

  請到台大圖書館網頁的電子論文服務下載論文格式範本，並修改成正確的格式，也可到此範本所在資料夾的cert.doc修改。當然你也可以利用 LaTeX 來編輯，你只要填好ntuvars.tex檔的資料，並去除在 thesis.tex 裡下面這行的註解符號`%`

```
%\makecertification
```

  編譯完後就可以產生審定書格式。口試通過後，請把已經簽名的審定書掃描成 pdf 檔，再取代原本的cert.pdf，即可放上已簽名的審定書。處理審定書出現的指令在 thesis.tex 裡

```
%----------- generate the certification ...
%\makecertification
%----------- includepdf by using package ...
\addcontentsline{toc}{chapter}{口試委員會審定書}
\includepdf[pages={1}]{cert.pdf}
```

- 浮水印：

  資料夾已經附上浮水印檔案了，若學校有更改，到請到台大圖書館網頁的電子論文服務下載pdf 格式的浮水印到此範本所在資料夾。若要開啟關閉浮水印功能，即自行刪去或加上下面位於thesis.tex指令的註解符號`%`

```
%\CenterWallPaper{0.174}{watermark.pdf}
%\setlength{\wpXoffset}{6.1725cm}
%\setlength{\wpYoffset}{10.5225cm}
```

- 單面印刷與雙面印刷：

  此範本為單面印刷，若論文頁數超過 80 頁，依規定需要用雙面印刷，此時只需把 thesis.tex 裡的

  ```
  \documentclass[a4paper, 12pt, oneside]{book}
  ```

  改 成

  ```
  \documentclass[a4paper, 12pt, twoside]{book}
  ```

- 如何加入附錄?

  在thesis.tex裡，依需求選擇 input 或 include，刪去% 符號來輸入附錄章節

  ```
  %----------- Input your appendix here  -----------
  %\input{AppendixA}
  %or %chapter cite  == \include
  %\include{AppendixA}
  ```

  在章節檔 AppendixA.tex 裡，開頭打

  ```
  \chapter{First appendix title}
  ```

  即可，以此類推。

- 系上規定論文圖表須全部放到最後獨立出來的章節，且章節不出現在目錄中：

  在thesis.tex裡，依需求選擇 input 或 include，刪去% 符號來輸入圖表章節

  ```
  %----------- Input your Figure chapter here  -----------
  %\input{EndFigTab}
  %chapter cite  == \include
  %\include{EndFigTab}
  ```

在章節檔EndFigTab.tex裡有範例和說明可供參考，要注意正文的圖表和附錄的圖表要分清楚，即在EndFigTab.tex內

```
\renewcommand{\thefigure}{\arabic{chapter}.\arabic{figure}}
\renewcommand{\thetable}{\arabic{chapter}.\arabic{table}}
%--- Input your main figures and tables here  ---
```

這幾行之後章節計數器格式已切換為 1...9，放正文的圖表，

```
\renewcommand{\thefigure}{\Alph{chapter}.\arabic{figure}}
\renewcommand{\thetable}{\Alph{chapter}.\arabic{table}}
%--- Input your appendix figures and tables here  ---
```

這幾行之後章節計數器格式已切換為 A...Z，放附錄的圖表。另外要取消圖表的浮動功能，才能讓圖表按照指令出現順序排好，即把平常使用的圖表指令

```
\begin{figure}[htb]
...
\begin{table}[htb]
```

改成

```
\begin{figure}[!]
...
\begin{table}[!]
```

剩下的只要注意章節圖表的計數器設定即可。`\ref` 和 `\label` 指令可以在此圖表章節與正文章節使用。

- 如果我想要修改 margin(文字邊界) 的話，可以從哪裡下手呢?
  請打開ntu.sty修改下面這行的上下左右參數即可：

```
\RequirePackage[top=3cm,left=3cm,bottom=2cm,right=3cm]{geometry}
```

- 我想引用 Twomey (1974): Pollution and planetary albedo 這篇論文，如何用 `\cite` 引用它的時候在內文顯示 Twomey (1974) [編號]？
  建議使用 natbib 套件，參考資料如下：
  LaTeX/Bibliography Management
  Overview of Bibtex-Styles
  Reference sheet for natbib usage

- XƎTEX：
  此範本中文字體使用XƎTEX 轉換，細節請參考Hitripod寫的XeTeX：解決 LaTeX 惱人的中文字型問題。

Bibliography

[1] M. Von Ardenne. Das elektronen-rastermikroskop. Zeitschrift für Physik A Hadrons and Nuclei, 109(9):553--572, 1938.

[2] G.K. Bennig. Atomic force microscope and method for imaging surfaces with atomic resolution, February 9 1988. US Patent 4,724,318.

[3] H. Raether. Surface plasmons. Springer-Verlag Berlin, 1988.

[4] C. Kittel and P. McEuen. Introduction to solid state physics, volume 7. Wiley New York, 1976.

[5] S.A. Maier. Plasmonics: fundamentals and applications. Springer Verlag, 2007.