creative commons

COMMONS DEED

저작자표시-비영리-동일조건변경허락 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.
- 이차적 저작물을 작성할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

동일조건변경허락. 귀하가 이 저작물을 개작, 변형 또는 가공했을 경우에는, 이 저작물과 동일한 이용허락조건하에서만 배포할 수 있습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

[UCI]I804:24011-200000242092

2011년 8월
석사학위논문

터보프롭 엔진의 역모델링과 인공지능 기법을 사용한 진단에 관한 연구

조선대학교 대학원

항 공 우 주 공 학 과

임 세 명

터보프롭 엔진의 역모델링과 인공지능 기법을 사용한 진단에 관한 연구

- Study on Fault Diagnostics of a Turboprop Engine Using Inverse Performance Model and Artificial Intelligent Methods -

2011 년 8 월 25 일

조 선 대 학 교 대 학 원

항 공 우 주 공 학 과

임 세 명

터보프롭 엔진의 역모델링과 인공지능 기법을 사용한 진단에 관한 연구

지도교수 공 창 덕

이 논문을 공학 석사학위신청 논문으로 제출함.

2011 년 4 월

조 선 대 학 교 대 학 원

항 공 우 주 공 학 과

임 세 명

임세명의 석사학위 논문을 인준함

위원장 국방과학연구소 책임연구원 오 성 환 (인)

위 원 (주)이지가스터빈 대표 기 자 영 (인)

위 원 조선대학교 교수 공 창 덕 (인)

2011 년 5 월

조 선 대 학 교 대 학 원

목 차

ABSTRACT

제 1 장 서 론 1
제1절 개 요 1
제2절 기술 현황 분석 4
1. 가스터빈 엔진 역 모델링 연구 4
2. 추진시스템 성능 모사 연구 5
3. 추진시스템 상태 진단 연구 8
제3절 연구 내용 및 범위 13

제 2 장 엔진 시험 장치를 이용한 역 모델링 연구 15
제1절 시험장치 구성 15
1. 대상 엔진 15
2. 엔진 센싱 홀 16
3. Test cell 16
제2절 데이터 계측 및 획득 시스템 17
1. Real-Time Monitoring System 17
2. 데이터 획득 시스템 18
제3절 실험 데이터를 이용한 성능선도 역생성 19
1. 성능선도 역생성 19
제4절 성능해석 결과 22

제 3 장 터보프롭엔진의 역 모델링 연구 26
제1절 대상 엔진 26
제2절 성능덱 데이터 분석 27

1. 입력조건 27
2. 성능덱 데이터 분석 28
제3절 구성품 성능선도 역생성 29
1. 압축기 29
2. 가스제너레이터 터빈 31
3. 동력 터빈 32
제4절 엔진 성능 모델링 및 생성된 구성품 성능선도 검증 33
1. RC-1 35
2. RC-3 36
3. RC-4 37
4. RC-5 38
5. RC-6 39
6. RC-7 40

제 4 장 터보프롭엔진의 인공지능 진단 연구 41
제1절 주요 구성품 손상에 따른 엔진 성능저하 특성 연구 41
1. 성능저하 원인에 따른 계측변수 분석 41
제2절 퍼지 및 신경회로망을 이용한 다중 손상 진단 46
1. 퍼지로직을 이용한 손상된 구성품 식별 46
2. 구성품 손상의 정량적 진단을 위한 신경회로망 설계 및 훈련 49
3. 제작된 인공지능 진단 프로그램의 검증 51
제3절 실제 엔진 시험데이터의 진단 56
1. 엔진 시험 데이터 분석 56
2. 인공지능 진단 프로그램을 이용한 엔진 시험 데이터의 진단 49

제 8 장 결 론 59

참 고 문 헌 60

부 록 67

LIST OF FIGURES

Fig. 1 Fault cause of America and Israel 1

Fig. 2 GASTURB, Gas turbine engine performance analysis program 6

Fig. 3 GSP, Gas turbine engine performance analysis program 7

Fig. 4 main/starting window, FRAME 8

Fig. 5 Neuro-fuzzy based FDI scheme 10

Fig. 6 Major elements of NPSS 11

Fig. 7 GE90-94B: Coupling of 3-D full engine model with 0-D cycle model 11

Fig. 8 Engine sensing hole 16

Fig. 9 Test cell 17

Fig. 10 Front Panel and Block Diagram 18

Fig. 11 Layout of signal line of data acquisition system 19

Fig. 12 Flow of component map generation using experimental test data 20

Fig. 13 Performance data of 90% RPM Compressor 21

Fig. 14 Numerical technique 22

Fig. 15 Generated maps of compressor and turbine 23

Fig. 16 Flow chart of performance simulation program using C++ 24

Fig. 17 Result of verification 24

Fig. 18 PT6A-67 Turboprop Engine 26

Fig. 19 Flow of Compressor Map Generation 30

Fig. 20 Generated Compressor Map of PT6A-67 Using Inverse Identification Method 31

Fig. 21 Generated HP Turbine Map of PT6A-67 Using Inverse Identification Method 32

Fig. 22 Generated LP Turbine Map of PT6A-67 Using Inverse Identification Method 33
Fig. 23 Flowchart of base performance simulation program 34
Fig. 24 Power Turbine Work 35
Fig. 25 SFC 35
Fig. 26 Power Turbine Work 36
Fig. 27 SFC 36
Fig. 28 Power Turbine Work 37
Fig. 29 SFC 37
Fig. 30 Power Turbine Work 38
Fig. 31 SFC 38
Fig. 32 Power Turbine Work 39
Fig. 33 SFC 39
Fig. 34 Power Turbine Work 40
Fig. 35 SFC 40
Fig. 36 Performance fallen rate of 45000ft, M0.3, RC4 Condition 45
Fig. 37 Flowchart of proposed diagnostic system 46
Fig. 38 MAMDANI type Fuzzy Inference System for isolating faulted components ... 48
Fig. 39 Feed forward Neural Network program using SIMULINK 49
Fig. 40 Back Propagation algorithms for training Neural Network 50
Fig. 41 Fuzzy-Neural Network Fault Diagnosis Program 51
Fig. 42 Results of faulted components quantified by Neural Network diagnostic program 55
Fig. 43 PT6A-67 Test Data(33min) 56
Fig. 44 PT6A-67 Test Data(3min) 57

LIST OF TABLES

Table 1 Fault rate of propulsion parts and Down times 2

Table 2 Specification of I-130 15

Table 3 Performance parameter values of compressor and turbine 22

Table 4 Design point performance of PT6A-67 turboprop 27

Table 5 Design point performance of PT6A-67 and PT6A-62 28

Table 6 Supposed performance fallen rate(Single fault) 42

Table 7 Supposed performance fallen rate(Multiple faults) 43

Table 8 Supposed performance fallen rate 43

Table 9 Considered component fault pattern cases 47

Table 10 Measured parameter change trends depending on component fault patterns ... 47

Table 11 Implanted fault values (IFV) of engine major components 51

Table 12 Measured parameter changes due to implanted faults (%) 52

Table 13 Results of faulted components isolated by Fuzzy Inference System (IFC: Input fault cases, OFC: Output fault cases) 52

Table 14 Test Data and Simulation Data 57

Table 15 Fault patterns of components 58

Table 16 Relationship between the chapters of the thesis and the published papers..... 61

NOMENCLATURE

F	Thrust
C_j	Exhaust Nozzle Exit Velocity
MFP	Mass Flow Parameter
PR	Pressure Ratio
ETA	Isentropic Efficiency
EGT	Exhaust Gas Temperature
M_a	Air Mass Flow
$COMA$	Compressor Mass Flow
$COEF$	Compressor Efficiency
FC	Fault Case
FIS	Fuzzy Inference System
$FFBP$	Feed Forward Back Propagation
GAs	Genetic Algorithms
GPA	Gas Path Analysis
$HTMA$	High pressure Turbine Mass Flow
$HTEF$	High pressure Turbine Efficiency
ITT	Inter Turbine Temperature (T5)
MF	Fuel Flow
MPC	Measured Parameter Change
NNs	Neural Networks
$PTMA$	Power Turbine Mass Flow
$PTEF$	Power Turbine Efficiency
RC	Flight Rating Code
RMS	Root Mean Square
TRQ	Torque
UAV	Unmanned Aerial Vehicle

ABSTRACT

Study on Fault Diagnostics of a Turboprop Engine Using Inverse Performance Model and Artificial Intelligent Methods

by Lim, Semyeong

Advisor : Prof. Kong, Changduk, Ph. D.

Department of Aerospace Engineering,

Graduate School of Chosun University

Recently, the health monitoring system of major gas path components of gas turbine uses mostly the model based method like the Gas Path Analysis (GPA). This method is to find quantity changes of component performance characteristic parameters such as isentropic efficiency and mass flow parameter by comparing between measured engine performance parameters such as temperatures, pressures, rotational speeds, fuel consumption, etc. and clean engine performance parameters without any engine faults which are calculated by the base engine performance model.

Recently, the expert engine diagnostic systems using the artificial intelligent methods such as Neural Networks (NNs), Fuzzy Logic and Genetic Algorithms (GAs) have been studied to improve the model based method. Among them the NNs are mostly used to the engine fault diagnostic system due to its good learning performance, but it has a drawback due to low accuracy and long learning time to build learning data base if there are large amount of learning data. In addition, it has a very complex structure for finding effectively single type faults or multiple type faults of gas path components.

This work builds inversely a base performance model of a turboprop engine to be used for a high altitude operation UAV using measured performance data, and proposes a fault diagnostic system using the base engine performance model and the artificial intelligent methods such as Fuzzy logic and Neural Network.

The proposed diagnostic system isolates firstly the faulted components using Fuzzy Logic, then quantifies faults of the identified components using the NN leaned by fault learning data base, which are obtained from the developed base performance model. In leaning the NN, the Feed Forward Back Propagation (FFBP) method is used.

Finally, it is verified through several test examples that the component faults implanted arbitrarily in the engine are well isolated and quantified by the proposed diagnostic system.

제 1 장 서 론

제 1 절 개 요

항공기용 추진 시스템은 타 추진 기관에 비해 손상이나 고장 시 사고 발생 리스크가 상당히 크다. 그렇기 때문에 항공기용 추진 시스템의 신뢰성과 내구성을 확보하는 것은 항공기 운용에 있어 매우 중요한 일이다. 이러한 신뢰성과 내구성, 그리고 유용성 등을 확보하기 위해 효율적인 정비에 대한 관심이 커지고 있는데 기존의 방식인 타임 테이블에 따른 정비는 불필요한 부분까지 모두 정비하는 등 시간이나 비용에 상당히 비효율적인 면이 있다. 따라서 최근에는 엔진의 상태를 실시간으로 감시하여 진단하여 필요한 부분만 정비를 수행하는 상태진단 방식이 활발히 연구되어 개발 및 적용이 이루어지고 있다.

본 연구 대상 엔진이 일반적으로 사용되는 무인항공기는 조종사가 부재하기 때문에 어떤 이상이 발생할 경우 사전에 이식된 프로그램에 의해 대처해야함으로 보다 정확한 성능 시뮬레이션과 진단시스템이 요구된다.

Figure 1은 미국의 무인항공기와 이스라엘 무인항공기의 주요 고장원인을 분석한 결과이다. 미국의 경우 1986년~2004년 말까지 누적비행시간 19만4000시간의 통계결과로 동력 및 프로펠러의 고장이 무인항공기 고장 원인의 38%로 차지해 가장 높은 고장원인이 되고 있는 것으로 나타났다. 이스라엘의 경우에도 동력 및 프로펠러 고장이 무인항공기 전체 고장의 32%를 차지하고 있다.

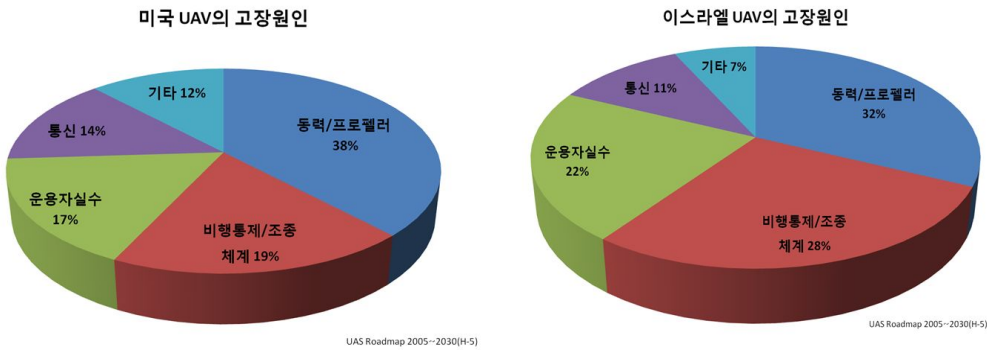


Fig. 1 Fault cause of America and Israel

표 1은 추진계통의 고장 원인을 분석한 것으로 엔진 자체의 주요 구성품 즉 압축기, 터빈 등에 손상이 발생하는 경우와 연료 시스템, 윤활 시스템과 같은 보기류의 고장, 로터 축, 베어링과 같은 기계적 부품의 마모로 손상패턴을 분류할 수 있다.

Table 1 Fault rate of propulsion parts and Down times

Outage Rate	%	Outage Total Down-Time	%
Control system	24.2	Generator	17.0
General gas turbine	19.2	Turbine	14.4
Fuel system	13.1	Compressor	12.0
Lubrication	11.1	General gas turbine	12.0
Combustion	8.3	Lubrication	11.5
Generator control	6.5	Power distribution	9.6
Power distribution	5.4	Combustion	7.9
Generator	4.8	Control system	7.5
Cooling system	3.8	Fuel system	4.7
Compressor	1.9	Generator control	2.2
Turbine	1.1	Cooling system	0.9
Exhaust system	0.6	Exhaust system	0.3

통계에서 알 수 있듯이 압축기, 터빈과 같은 추진시스템의 주요구성품의 경우 손상 발생 빈도는 낮으나 이상이 발생하였을 경우 엔진 전체 성능에 미치는 영향이 치명적이며 수리 및 교체를 위한 비용과 Down-time도 다른 부품에 비해 매우 크다. 이는 시스템의 유용성을 감소시키는 주요 원인이 된다.

압축기 블레이드에 오염이 발생하여 3 %의 유량감소와 1 %의 효율이 감소하게 되면, 약 4.9 %의 추력 감소와 1.47 %의 비 연료소모율 증가를 가져오게 된다. 터빈의 부식은 효율 감소와 함께 블레이드의 내구성을 저하시키는 원인이 되고, 부품의 마모는 효율 감소 등 성능저하뿐 아니라 항공기 전체의 안정성과 신뢰성을 감소시키게 된다. 로터 축, 베어링과 같은 기계적 손상은 엔진 수명과 직결되는 것으로 상태진단을 통한 적절한 관리, 운용으로 엔진의 수명 연장과 함께 경제적 손실을 최소화 할 수 있다. 연료 시스템, 윤활 시스템과 같은 보기류의 이상은 연료소모율의 증가를 가져오게 되며, 이것이 지속되었을 경우 엔진의 성능에도 영향을 미치게 된다.

이러한 추진시스템의 고장을 최소화하기 위해서는 지상정비창과 연계된 온라인 상태진단 시스템의 구축이 요구된다. 상태진단 시스템은 압축기 및 터빈과 같은 주요구성품의 성능진단, 보기류 및 부품의 상태 감시를 통해 시스템의 고장을 최소화하고 고장 후

정비에 소요되는 시간과 비용을 최소화 할 수 있도록 함으로써 최적의 상태에서 운용될 수 있도록 하는 것이다.

고도 40000ft 이상의 고공에서 장시간 운용되는 무인항공기의 추진시스템은 무인으로 제어됨에 따라 요구 출력 및 운용한계의 자동제어가 요구되며, 이를 위해 정확한 엔진 모델 및 고고도 작동특성에 대한 연구가 필요하다. 또한 열악한 환경에서 운용되는 엔진의 신뢰성과 내구성을 확보하기 위해서는 모델기반 상태모니터링 연구가 필요하다.

이러한 모델기반 상태모니터링을 위해서는 엔진의 기준 성능모델링이 선행되어야 하며 이는 엔진의 실제 계측 데이터와 시뮬레이션 결과를 비교 분석하여 이상 유무를 판단하는 방법으로, 엔진 전체성능에 크게 영향을 주는 압축기 및 터빈뿐 아니라 기타 구성품의 정확한 성능모델링이 상태진단의 신뢰성을 보장한다.

엔진의 성능을 정확하게 모사하기 위해서는 주요구성품의 시험을 통해 얻어진 성능선도가 매우 중요하다. 하지만 이는 엔진 제작사의 고유 자산으로 공개가 되지 않으므로 구매자는 다양한 방법으로 역모델링 방법을 연구해야 한다.

본 연구에서는 무인기용 터보프롭 엔진의 역 모델링 및 상태진단 연구를 수행하였다.

선행연구로서 마이크로 터보제트 엔진 시험장치를 이용하여 역모델링 연구를 수행해 아였으며 엔진 성능데이터를 활용하여 엔진 및 구성품의 성능모델링을 수행하고 엔진 모델을 이용하여 성능데이터베이스를 구축하였다. 이렇게 구축된 데이터베이스를 이용하여 퍼지룰을 생성하고 또한 신경회로망을 훈련시켜 인공지능 진단프로그램을 구성한 후 검증하였다. 마지막으로 실제 엔진 데이터를 진단프로그램으로 진단을 수행하였다.

제 2 절 기술 현황 분석

1. 가스터빈엔진 역 모델링 연구

신뢰성 있는 엔진 모델링을 위해서는 주요구성품의 성능특성을 담은 성능선도가 필수적이나 이는 제작사의 고유자산을 공개가 되지 않으므로 이를 역으로 구하기 위한 연구가 진행되어 왔다.

Hormouziadis와 Herbig(1974)[1]은 압축기 성능선도의 다양한 양상을 표현할 수 있는 속도 곡선과 효율 등고선을 기술하기 위해 유리함수와 타원 근사를 도입하였다. Dobryanskii와 Martyanova(1989)[2]는 Look-up 테이블을 사용하여 성능선도 데이터를 저장하고 선형 또는 Lagrangian 보간법을 이용하여 성능선도의 임의 점에서 변수의 값을 정의하는 방법을 제안하였다. El-Gammal(1991)[3]은 압축기의 선형모델을 위한 기준과 알고리즘을 개발하였는데 이 방법은 잘 알려진 성능 데이터로부터 가장 적당한 모델을 객관적으로 선택할 수 있도록 한다. Sieros(1997)[4] 등은 해석적 함수를 사용하여 압축기와 터빈 성능선도를 비선형으로 표현할 수 있게 하였으며, Orkisz와 Stawarz(1997)[5]는 축류 압축기와 터빈 성능선도의 변수들(유량, 압력비, 효율, 회전수) 사이의 관계를 함수적으로 표현하는 방법을 제시하였다. Kurzke(2000)[6]등은 많은 압축기 성능선도를 통계적으로 해석하고 각 성능선도에서 효율이 최고인 영역, 유량-속도의 관계, 속도 선도의 형상 등을 정의하여 새로운 성능선도를 구성하는 방법을 제시하였다. 그러나 이 방법은 성능선도에 대한 방대한 자료가 요구됨으로 보유하고 있는 성능데이터 자료가 적은 경우에는 이용하기 어려운 단점이 있다.

국내에서도 공창덕(2006년)등이 엔진 시험데이터를 이용하여 시스템 식별 기법을 이용하여 압축기와 터빈의 성능선도를 역으로 구성하는 방법을 제시하였으며 (2007년)에는 제작사에 제공하는 성능덱 데이터로부터 유전알고리즘을 이용하여 성능선도를 역으로 구성하는 방법을 제시하였다. 시험장치 및 성능덱 데이터를 이용하여 역으로 구성품 성능선도를 식별하는 방법은 2007년 국내특허(제 10-0788212호)로 등록되었다.

2. 추진시스템 성능 모사 연구

1970년대 초부터 비용이 많이 들고 위험도가 높은 엔진 시험을 대체하기 위한 성능해석에 관한 연구가 활발하게 진행되어왔으며 컴퓨터의 발달과 함께 많은 발전을 이루고 있다. Seller and Daniele(1975)[8]은 다양한 형태의 Turbojet와 Turbofan 엔진의 정상상태 및 천이성능 해석이 가능한 DYNGEN 코드를 개발하였으면 현재에도 참고자료로 활용되고 있다. MacMillan(1974)[9]는 각 구성품별 성능해석 코드를 블록화 하여 거의 모든 종류의 가스터빈에 대한 정상상태 성능해석이 가능한 TURBOMACH 프로그램을 개발하였는데, Cranfield University(U.K)에서 현재까지도 보완, 수정하여 이용하고 있다. Fishbach and Caddy (1975)[10]은 NNEP를 개발하여 엔진 무게 해석, 장착 손실 해석 등을 포함하여 보다 세밀한 해석이 가능하도록 하였다.

Palmer and Yan(1985)[11]은 TURBOMACH 프로그램을 보완하여 TURBOTRANS 프로그램을 개발하였으며 주 연소기 유량, 후기연소기 유량, 바이패스 및 가스발생기 노즐 면적 제어 알고리즘이 추가되었다. El-Masri (1988)[12]가 개발한 GASCAN 프로그램은 열역학, 열전달, 유체역학을 공식화하여 보다 세밀한 열역학적 현상을 해석할 수 있도록 하였다. Schobeiri et al (1994)은 GETRAN 프로그램에서 압축기와 터빈의 성능 모사에서 비 단열과 단열 효과를 계산할 수 있는 알고리즘을 제안하였으며 Bettocchi et al(1996)[13]과 Crosa et al (1998)[14]는 SIMULINK를 이용하여 GUI 성능해석 프로그램을 개발하였다. J. Kurzke (1998)는 임의의 가스터빈 엔진의 정상상태 및 천이성능 해석을 위한 GUI 프로그램인 GASTURB를 개발하여 현재 상용프로그램으로 널리 이용되고 있다.

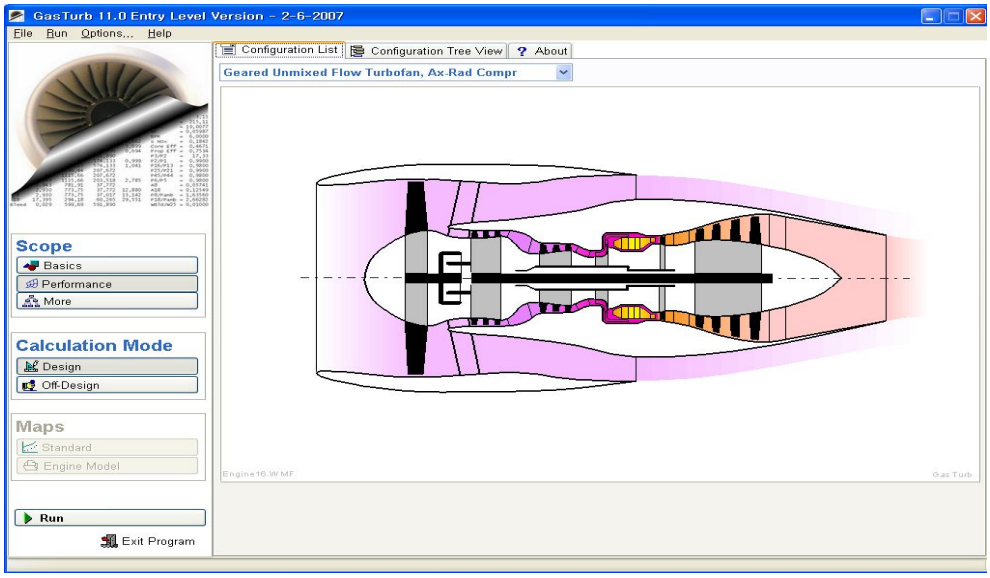


Fig. 2 GASTURB, Gas turbine engine performance analysis program

NLR(National Aerospace Laboratory, The Netherlands)에서는 군용 Turbojet 엔진과 Turbofan 엔진의 제어시스템 손상 효과를 해석할 수 있고, 성능데이터를 이용한 Health Monitoring이 가능한 GSP(Gas turbine Simulation Program)을 개발하여 상용화 하였으며, Reed and Abdollah(1999)는 Java를 기반으로 한 ONYX system을 개발하였다. 이 프로그램은 Object-oriented 시스템으로 Web 상에서 구동이 가능하다.

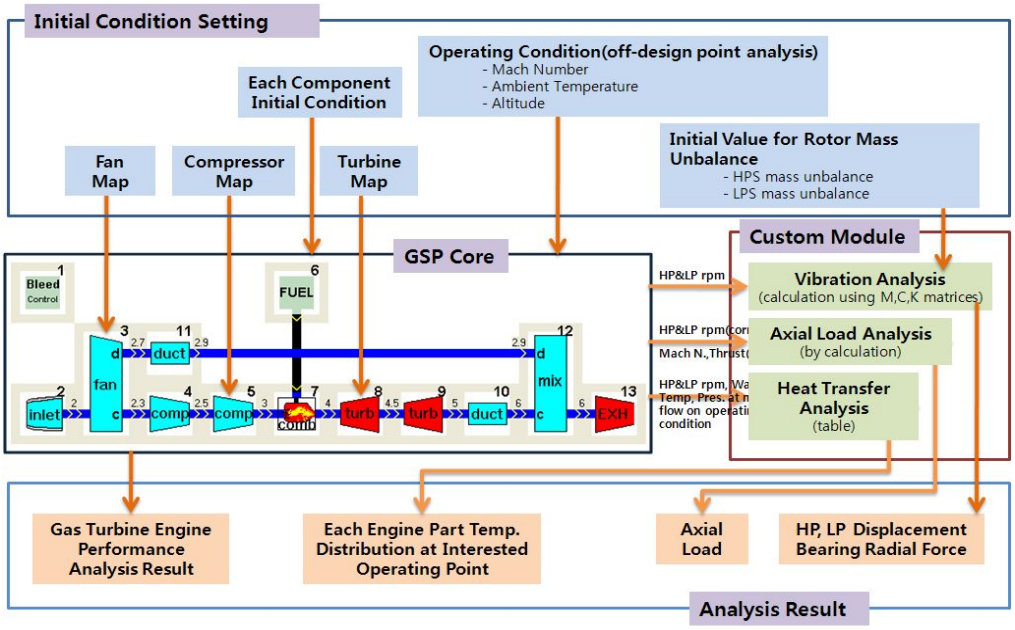


Fig. 3 GSP, Gas turbine engine performance analysis program

AFFDL(Air Force Flight Dynamics Laboratory)의 기술보고서에서는 장착에 의한 흡입구와 배기노즐의 성능데이터를 포함한 성능해석 프로그램 개발에 관한 내용을 포함하고 있다. 또한 Cranfield의 “Propulsion System Performance and Integration” Short Course 자료에는 장착에 따른 흡입구과 배기노즐의 성능에 대해 제시되어 있으며, ESDU 기술보고서는 장착에 의한 손실, 비행속도 및 대기조건의 변화가 발생시키는 성능손실에 대해 분석하고 있다.

Baxevanidis(2001)는 고고도에서 장시간 운용되는 무인항공기의 성능에 대한 연구를 수행하였으며, Maggio et al(2002)등은 추진시스템의 신뢰성 평가에 대한 연구 결과를 발표하였다. 또한, Milanes et al(2004) 등은 연소기의 내구성을 강화하기 위한 방안에 대해 연구를 수행하고, Veer et al(2004)등은 가스터빈 엔진의 신뢰성 및 유용성 평가와 함께 상태를 예측할 수 있는 FRAME을 개발한 바 있다.

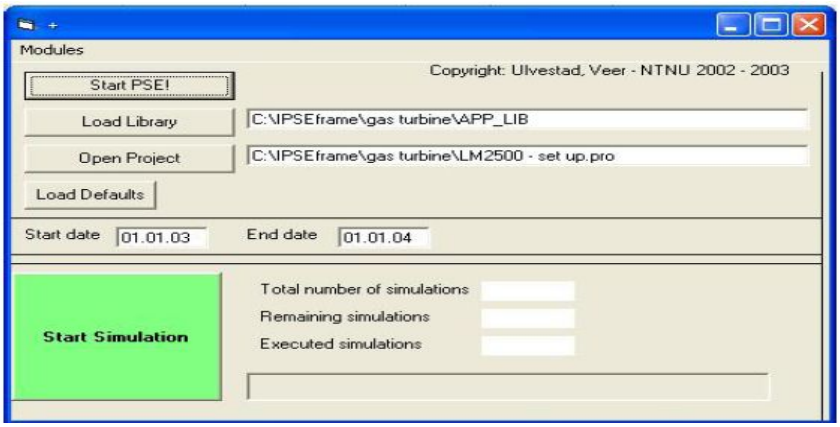


Fig. 4 main/starting window, FRAME

국내에서도 1990년대에 들어 성능모사에 대한 중요성을 인식하고 그에 대한 연구가 진행되고 있으나 선진국에 비해 아직 그 기술력이 미흡한 실정이다. 1990년대 초 국방과학연구소에서는 소모성 소형 터보엔진 개발을 수행하며 자체개발 성능해석 코드 및 DYNGEN 코드를 이용한 터보제트 엔진의 정상상태 및 천이성능해석에 대한 연구가 수행되었으며 2000년에는 KT-1 기본훈련기 개발과 연관하여 터보프롭 추진시스템의 정, 동적 성능모사 프로그램 개발에 관한 연구를 수행하였으며 흡입구등의 장착손실을 고려한 성능해석의 기초연구가 수행되었다.

3. 추진시스템 상태 진단 연구

가스터빈 엔진의 신뢰성을 높이기 위해 운용 상태를 실시간으로 감시하고 진단할 수 있는 시스템에 관한 연구와 개발이 연구기관 및 관련 산업체를 중심으로 이루어지고 있으며 최근 상용화된 시스템들이 운용 중에 있다.

모델 기반 온라인 상태감시에 대한 연구는 1974년 Urban이 공기와 연소가스가 통과하는 구성품의 효율 감소와 질량유량의 변화량으로 엔진성능을 예측하는 방법인 선형 GPA 이론을 제시하며 본격화 되었다. 이 방법은 고장 시 엔진 전체의 성능과 운용에 치명적인 영향을 미치는 주요구성품(압축기, 연소기, 터빈 등)의 상태를 정량적으로 분석하여 주요구성품의 상태를 진단하고 고장의 원인을 예측한다. 그 후로 GPA 기법의 성능을 향상시키기 위한 연구들이 활발히 진행되어 Grewal(1988)[15]는 선형 GPA 이론

을 이용하여 다양한 성능저하의 원인을 분석하고 모델링 하였으며 Diakunchak(1992)[16]은 산업용 가스터빈의 성능저하 실험을 통해 고장 원인과 엔진 상태의 관계를 연구하였다. House(1992)[17]은 터보축 엔진을 위한 선형 GPA 프로그램을 개발하였으며 Esher(1995)[18,19]는 선형 GPA 기법의 한계를 극복하기 위한 비선형 GPA 기법을 제안하고 임의 엔진에 적용할 수 있는 비선형 GPA 기법을 이용한 객체지향 프로그램을 개발하였다. MacDonald(1993)는 GE-F404 엔진의 EHM을 위한 동적거동 모사에 관한 연구를 수행하였으며 천이상태에서의 GPA(Gas Path Analysis)를 수행하고 그 결과를 제어기와 연계하는 알고리즘을 제시하였다. 성능진단에 인공지능 이론을 적용함으로써 진단 시스템의 성능을 향상시키기 위한 연구도 다수 진행되었는데 Vivian(1993)은 GPA 기법에 KBS(Knowlege Based System)를 적용한 가스터빈 엔진의 진단 전문가 시스템 개발에 관한 연구를 수행하였으며 Boucherma(1994)는 신경회로망 이론을 이용한 터보 제너레이터의 손상탐지와 진단에 관한 연구를 수행하였다. Zhou(1998)는 퍼지논리를 이용하여 엔진 손상을 탐지하고 신경회로망 이론을 적용하여 진단과 손상 패턴을 분류하는 지능형 진단 시스템의 개발에 관한 연구 결과를 발표하였으며 Patel(1998)은 가스터빈 엔진의 모델링을 SIMULINK를 이용하여 수행하고 신경회로망을 통해 엔진 고장을 진단하는 연구를 수행하였다. Kim(1998)[20]은 신경회로망 이론을 적용함으로써 GPA의 가스 경로 해석 방법을 이용할 때 발생하는 다수의 측정 변수 선정 문제를 해결하여 보다 적은 계측변수로 원하는 진단 결과를 얻고자 하였다. DePold(1999)는 진단을 위해 신경회로망을 이용하고 엔진의 상태를 예측할 수 있는 전문가 시스템에 관한 연구를 수행하였다. Merrington(1994)[21]은 Model-Based 기법을 이용하여 엔진 고장을 진단하였으며 Sampath(2002)는 유전자 알고리즘을 이용하여 엔진 손상을 진단하는 방법을 제안하였다. Palade(2002)등은 산업용 가스터빈 엔진을 SIMULINK를 이용하여 모델링하고 Neuro-Fuzzy 기법을 이용하여 손상을 탐지하고 분리하는 FDI(Fault Detection and Isolation) 시스템에 관한 연구를 수행하였다. 이 연구에서는 신경회로망의 학습 및 시스템 식별 능력과 Fuzzy 시스템의 다양한 환경에서 보다 인간의 지능에 가깝게 지식을 표현하는 기능을 접목하여 진단 시스템의 기능이 향상됨을 보여주었다.

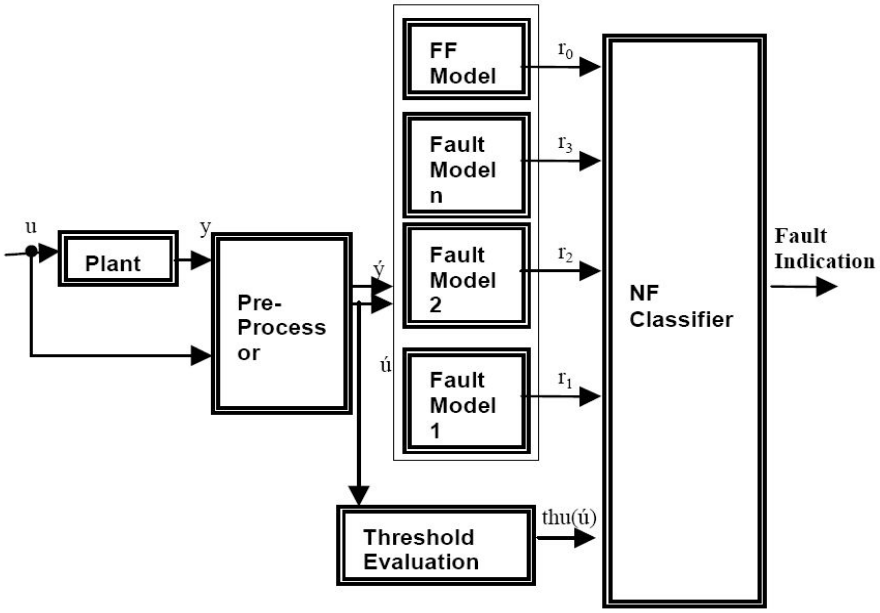


Fig. 5 Neuro-fuzzy based FDI scheme

2000년 초부터 미국의 NASA Glenn Research Center를 중심으로 추진 시스템의 통합 3D (three-dimension) 해석이 가능한 NPSS (Numerical Propulsion System Simulation) 개발 프로젝트가 시작되었다. Figure 6과 같이 NPSS는 객체지향프로그래밍(object oriented programming)을 사용하여 고 신뢰 해석(high-fidelity analysis)과 수치 확대 기법(numerical zooming technique)이 가능한 프로그램으로써 현재 미국 우주항공 표준 시뮬레이션으로 사용되고 있다. 이에 유럽에서도 추진시스템의 개발 시간과 비용을 절감하기 위한 유럽 가스터빈 표준 시뮬레이션인 PROOSIS(PRopulsion Object Oriented SImulation Software) 개발 프로젝트를 시작하여 현재 수정 보완 중에 있다.

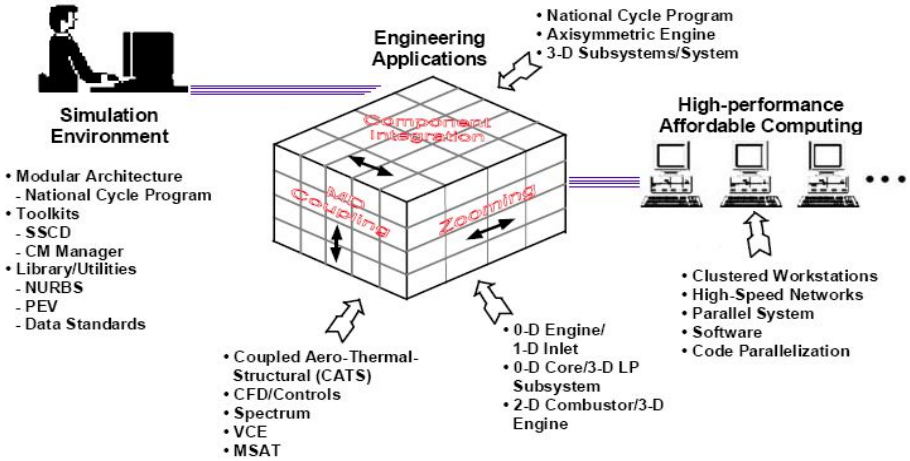


Fig. 6 Major elements of NPSS

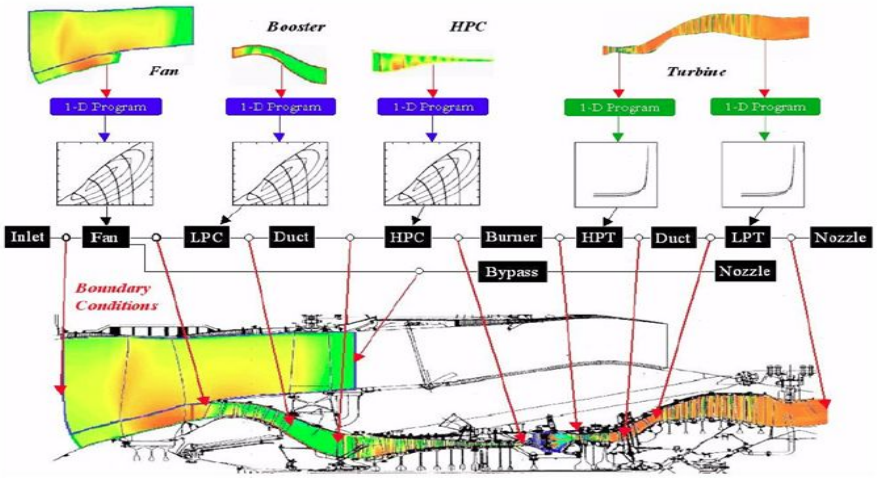


Fig. 7 GE90-94B: Coupling of 3-D full engine model with 0-D cycle model

엔진의 상태를 실시간으로 감시하고 고장시 안전한 운용을 위한 제어 로직에 관한 연구도 진행되었는데, Cue(1991)등은 CF-18 항공기를 위한 엔진의 성능 진단과 고장 시 대체 로직의 가동을 비행 상태에서 실시간으로 수행하는 IECMS(In-flight Engine Condition Monitoring System)의 개발에 관한 연구를 발표하였으며 Kerr(1992)등은 Kalman Filter 알고리즘을 이용하여 가스터빈 엔진의 손상 여부를 실시간으로 예측하는

방법을 제안하였다.

국내에서도 최근 가스터빈 엔진 상태진단 시스템에 관한 관심이 증가하며 항공기 및 가스터빈 엔진 개발 프로젝트와 함께 연구가 증가하고 있다. 1990년대에는 압축기와 터빈의 성능저하를 진단할 수 있는 GPA(Gas Path Analysis) 알고리즘에 관한 연구가 많이 진행되었으며 2000년대에는 한국형 헬기개발, 스마트무인기 개발, 5MW급 산업용 가스터빈 개발과 관련하여 신경회로망, 퍼지알고리즘 등 인공지능 이론을 적용한 고장진단 기술개발에 대한 연구가 이루어지고 있다.

제 3 절 연구 내용 및 범위

본 연구에서는 무인기용 터보프롭엔진의 역모델링과 이렇게 제작한 기준 엔진 모델을 이용하여 인공지능 진단 프로그램을 제작하였다.

먼저 역모델링 연구 기법 연구와 실제 시험 데이터 활용 방안 연구를 위해 선행 연구로서 마이크로 터보제트 엔진의 시험 장치를 구성하고 이를 이용하여 성능시험을 수행해 얻어진 데이터를 기반으로 마이크로 터보제트 엔진의 구성품 성능선도 역생성과 기준 엔진 모델을 제작하였다. 그리고 다시 성능 시험을 수행하여 얻어진 성능 데이터와 제작된 엔진 모델 시뮬레이션 결과를 비교하여 검증하였다.

우선 정확한 성능모델링과 해석을 위해서는 주요 구성품의 성능선도 확보가 필수적이다. 그러나 성능선도는 제작사의 고유자산으로 공개가 되지 않아 구매자 또는 연구자 입장에서는 실제 엔진의 성능선도를 사용하여 모델링을 하는 것은 불가능하며 일반적으로 공개된 유사 엔진의 성능선도를 축척하여 이용한다.

일반적으로 이용되어오던 성능선도 축척방법은 제작사에서 제공한 설계점 성능을 기준으로 얻은 축척 비율로 전 영역의 성능선도를 축척한다. 이렇게 축척된 성능선도를 이용하는 경우 설계점 부근의 성능은 실제 엔진과 유사하나 설계점에서 멀어질수록 실제 엔진의 성능과는 동떨어진 해석결과를 보인다는 것이다.

이에 공창덕 등은 성능선도 상의 한 운용점(설계점)을 기준으로한 축척이 아닌 여러 운용점에서 얻은 축척 비율로 성능선도를 식별하는 System Identification Method와 유전자 알고리즘을 통해 운용환경 변화에 따른 성능선도를 식별하는 Genetic Algorithm Method, 두 방법을 모두 사용하여 성능선도를 식별하는 Hybrid Method를 연구하였다.

본 연구에서는 System Identification Method를 개선하여 Inverse Identification Method를 제안하였다. 본 연구에서 사용한 이 방법은 성능데이터로부터 역으로 계산하여 성능선도를 식별하는 방법으로서 비교적 높은 정확도를 확보할 수 있다.

이렇게 선행 연구에서 얻어진 연구 결과와 경험을 바탕으로 1차적으로 터보프롭 엔진의 성능덱을 이용하여 다양한 조건에서의 성능 데이터를 축적하였다. 이렇게 축적된 성능데이터에서부터 열역학적 계산과 Inverse Identification Method를 이용하여 대상 엔진의 성능선도를 식별하였다.

이렇게 식별된 성능선도를 검증하기 위해 C++을 이용하여 추진시스템의 성능모델링을 수행하였다. 흡입구, 압축기, 연소기, 압축기터빈, 동력터빈 등 각 구성품을 모델링하였고 운용점을 가정하여 동축으로 연결된 압축기와 터빈 사이의 일적합을 만족하는 운용점을 찾도록 하였다. 그리고 기준 엔진 모델은 계산 결과와 성능덱의 결과를 비교하여 검증하였다.

그 후에 제작된 기준 엔진 성능 모델을 이용하여 성능 저하에 따른 다양한 조건에서의 성능데이터베이스를 구축하였다.

2차적으로 구축된 성능데이터베이스를 이용하여 퍼지룰을 생성하고 신경회로망을 훈련시켜 퍼지-신경망 인공지능 진단 프로그램을 제작하였다.

진단 프로그램은 정성적인 고장진단을 하는 퍼지로직 부분과 정량적인 고장진단을 하는 신경회로망 부분으로 나뉘며 동력터빈 입구 온도, 배기가스 온도, 연료 유량, 토크를 입력 값으로 압축기, 고압터빈, 저압터빈의 유량함수와 효율이 출력 값이다.

7개로 정의한 Fault Case에 해당하는 경우가 없을 경우 클린엔진으로 판별하게 된다.

이렇게 제작된 진단 프로그램은 강제로 손상을 준 값으로 고장 진단을 잘 수행하는지 검증하였으며 실제 엔진 시험 데이터를 이용하여 진단을 수행하였다.

제 2 장 엔진 시험 장치를 이용한 역 모델링 연구

제 1 절 시험장치 구성

1. 대상 엔진

시험장치의 구성에는 I-Complete 사의 I-Jet 130 엔진을 사용하였다. i-Jet 130은 추력 137.2N 급의 소형 터보제트 엔진으로 주로 Radio Control Micro Aircraft의 추진장치로 사용되고 있으며 1단 원심 압축기, 역류 애눌러형 연소실, 1단 축류 터빈으로 구성되어 있다. 또한 연료 공급을 위한 연료펌프, 연료필터, 연료밸브 및 점화플러그, Starter Motor 등의 accessory가 있으며 ECU에 의해 자동으로 제어된다. 원활한 시동을 위해 시동 시에는 프로판 가스가 사용되고 idle 이상 정상 작동 시에는 케로신 연료가 공급된다. Table 2는 I-jet 130엔진의 주요 제원이다[22].

Table 2 Specification of I-130

Type	**Turbojet**
Compressor	1 stage centrifugal
Combustion Chamber	Reverse flow annular
Turbine	1 stage axial
Thrust (max.@126000 rpm)	137.2N(30.86 lbf)
Thrust (idle@32000 rpm)	4.02N(0.9 lbf)
Exhaust Gas Temperature(max.)	853 K
Fuel Consumption	0.139 g/N/h
Fuel	Kerosene, Jet A1
Start Gas	Propane
Lubrication Oil	Mobil Jet2, Exxon 2380
Fuel-oil Mixing Ratio	20:1
Dimensions	115 mm ∅ x 320 mm L

2. 엔진 센싱 홀

공기속도, 압축기 입/출구와 터빈 출구의 온도 및 압력을 측정하기 위하여 Figure 8과 같이 Sensing hole을 만들고 압력 측정용 튜브, 온도 센서, 공기 속도 센서를 장착하였다[23],[24],[25].

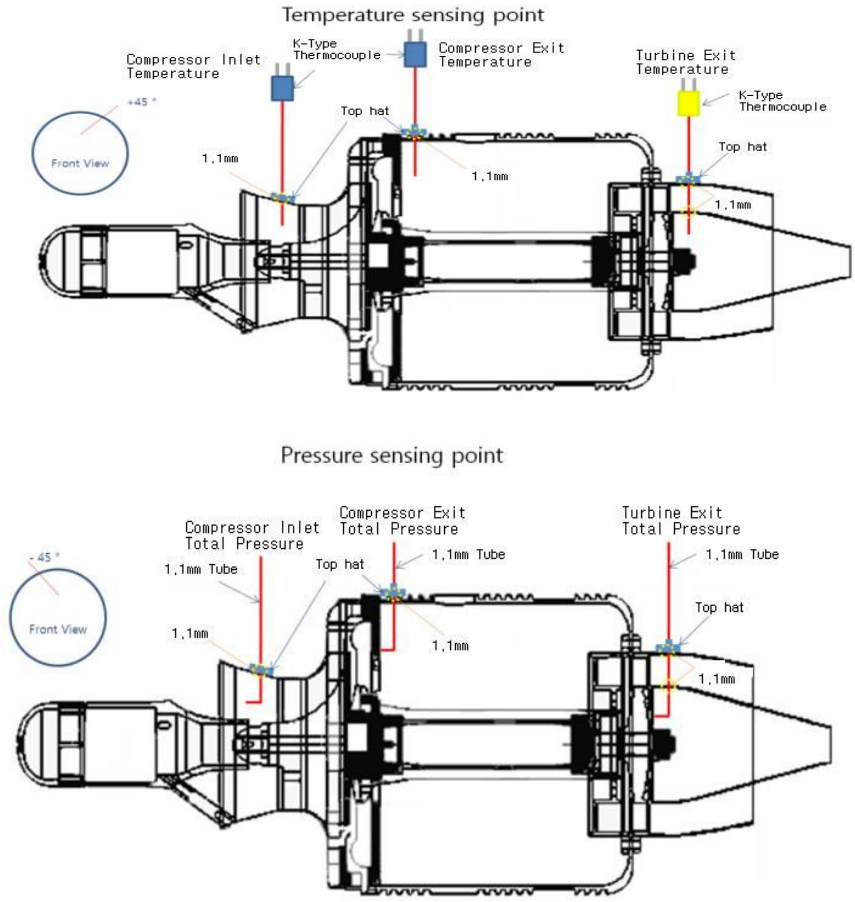


Fig. 8 Engine sensing hole

3. Test cell

테스트 셀은 추력대를 지지해주고, 터보제트 엔진 시험 중 발생할 수 있는 위험으로부터 시험자를 보호해주는 안전 케이스 역할을 한다. 또한 고온의 배기가스를 안전하게 배출하기 위한 배기구가 장착되어 있으며 데이터 모니터링 및 운용이 쉽고 편리하도록 설계 및 제작하였다. Figure 9는 제작 완료된 테스트 셀이다.

Fig. 9 Test cell

제 2 절 데이터 계측 및 획득 시스템

데이터 모니터링 및 획득 장치는 NI(National Instruments)사에서 개발한 LabVIEW를 사용하였다.

1. Real-Time Monitoring System

실시간 모니터링 화면은 사용자가 엔진 테스트 중 실시간으로 계측되는 데이터를 한눈에 쉽게 읽고 확인할 수 있도록 배치하였다. 시험 장치에서 모니터링 및 획득되는 데이터는 압축기 입구 온도와 압력, 압축기 출구 온도와 압력, 터빈 출구 온도와 압력, 연료유량, 추력 값이다. Figure 10은 LabVIEW를 이용하여 구성한 프로그래밍한 프론트 패널과 그 블록 다이어그램이다.

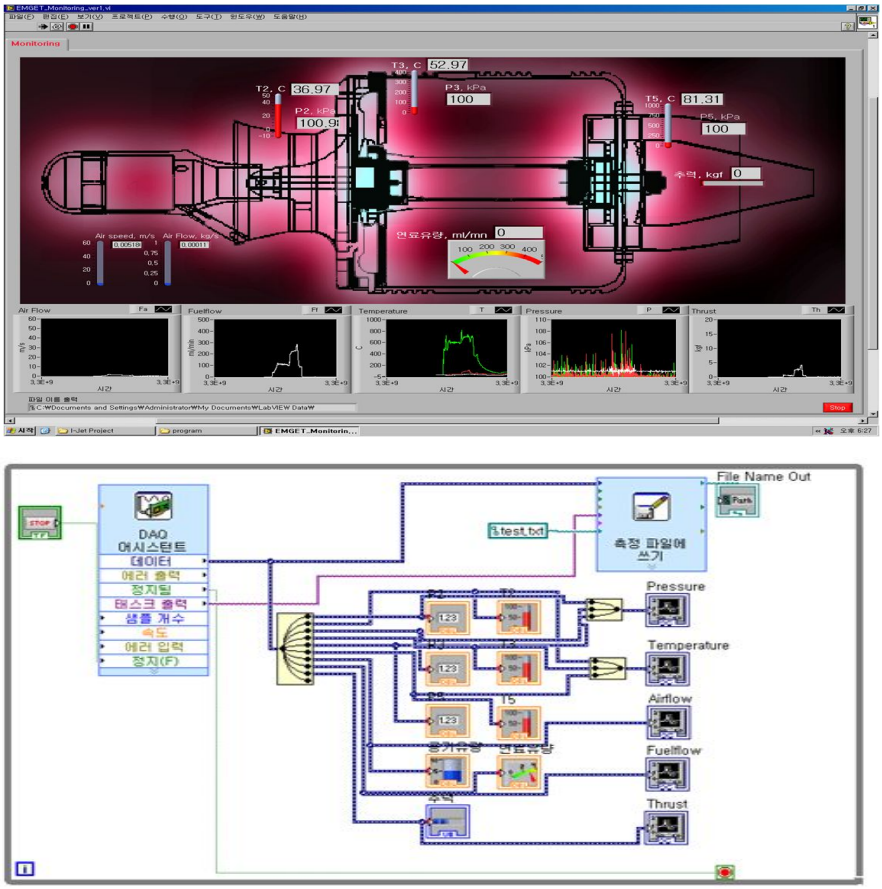


Fig. 10 Front Panel and Block Diagram

2. 데이터 획득 시스템

데이터 획득 장치로는 NI CompactDAQ 하드웨어를 사용하였으며 시험장치에서 계측되는 센서의 아날로그 신호를 간단한 플러그 앤 플레이로 빠르고 정확한 측정이 가능하다. Figure 11은 DAQ 장비를 사용한 데이터 획득 시스템의 배선도이다.

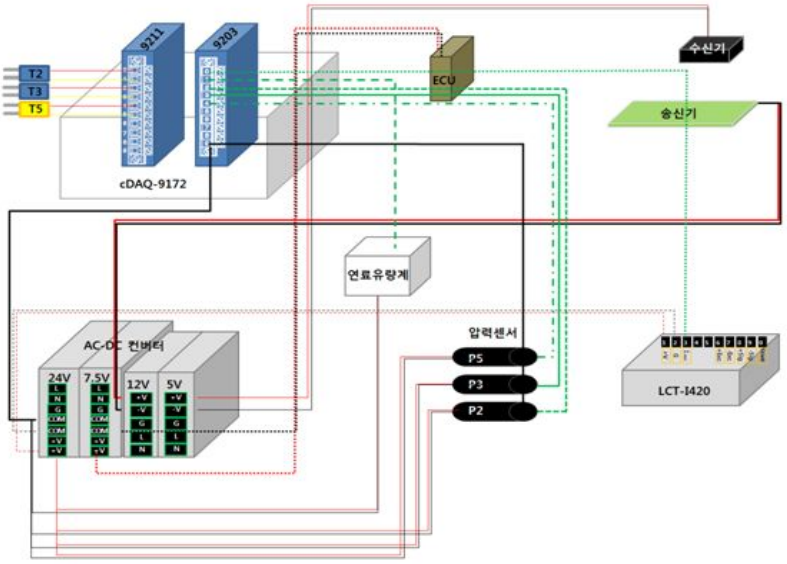


Fig. 11 Layout of signal line of data acquisition system

제 3 절 실험 데이터를 이용한 성능선도 역생성

실험은 i-Jet 130엔진의 50%, 60%, 70%, 80%, 90%에 해당하는 RPM별로 수행하였다. 데이터 획득은 각 RPM별로 10Hz로 30초씩 이루어졌다.

1. 성능선도 역생성

실험 데이터를 이용한 구성품 성능선도 축척을 위한 계산 절차는 Figure 12와 같다. 먼저, 각 구성품 축척인자들을 결정하기 위해 시험장치의 실험으로부터 얻어진 각 RPM별 성능 데이터인 온도, 압력, 연료유량, 추력과 식 (1)과 (2)를 이용하여 배기제트 속도(Cj)와 공기유량(WA)을 계산한다[26].

$$Cj = \left\{ 2C_P \eta_j T_4 \left[1 - \left(\frac{1}{P_5/P_a} \right)^{\frac{\gamma-1}{\gamma}} \right] \right\}^{\frac{1}{2}} \tag{1}$$

$$F = WA[(1+f)Cj] \tag{2}$$

C_P : specific heat at constant pressure,

η_j : nozzle efficiency assumed as 0.98,

T_4 : turbine exit temperature,

P_4 : turbine exit pressure, F: thrust.

P_a : ambient pressure, γ : specific heat ratio,

WA : air flow rate, f : fuel flow rate,

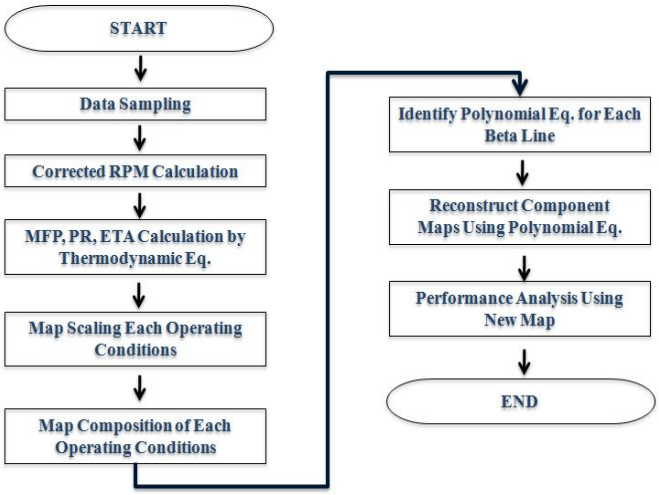


Fig. 12 Flow of component map generation using experimental test data

이렇게 획득된 배기제트 속도(Cj)와 공기유량 데이터(WA)로부터 열역학적 방정식을 이용하여 역으로 압축기와 터빈의 압력비, 유량함수 등의 성능데이터를 계산해 낼 수 있다. Figure 13은 90% RPM에서의 압축기 성능 데이터이다.

$$\mathrm{Pr}_c = \frac{P_3}{P_2} \tag{3}$$

$$MFP_c = \frac{\dot{m}\sqrt{T_2}}{P_2} \tag{4}$$

$$\eta_c = \frac{T_2}{T_3 - T_2}\left[(\mathrm{Pr})^{\frac{\gamma-1}{\gamma}} - 1\right] \tag{5}$$

$$\mathrm{Pr}_t = \frac{P_5}{P_4} \tag{6}$$

$$MFP_t = \frac{\dot{m}\sqrt{T_4}}{P_4} \tag{7}$$

$$\eta_t = \frac{T_4 - T_5}{T_4\left[1-\left(\frac{1}{\mathrm{Pr}_t}\right)^{\frac{\gamma-1}{\gamma}}\right]} \tag{8}$$

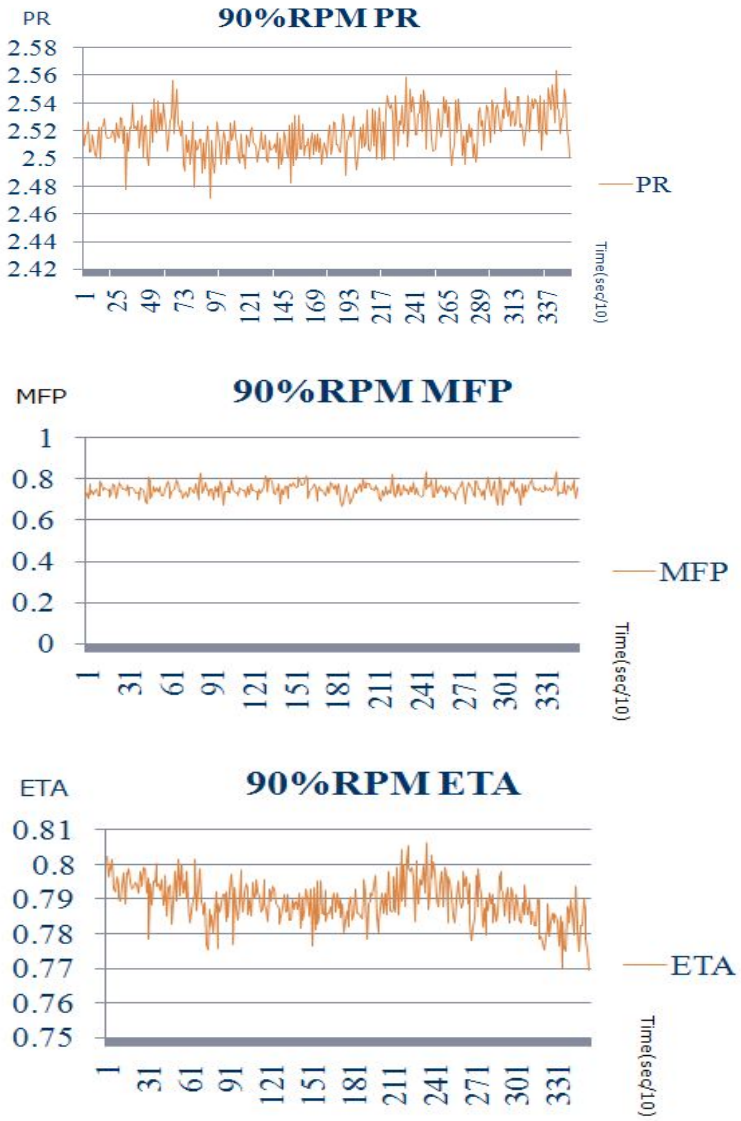


Fig. 13 Performance data of 90% RPM Compressor

이렇게 획득된 구성품 성능 데이터는 수치해석적 기법으로 정리하여 각 조건에서의 설계점 성능 값으로 적용했다. 수치해석적 기법은 노이즈 제거와 경향분석을 위해 획득된 데이터들의 최대점과 최소점 사이를 다섯 구간으로 구분하여 그 구간들 중 가장 많

은 데이터가 집중되어있는 구간을 선별하여 평균값을 취하는 방법을 사용하였다. Figure 14는 수치해석 방법에 대한 설명을 그림으로 나타낸 것이다.

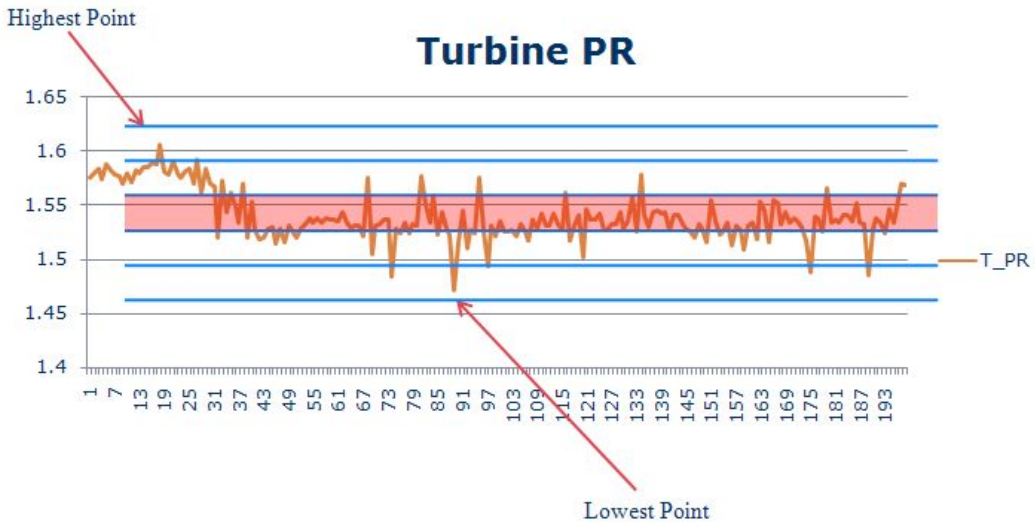


Fig. 14 Numerical technique

Table 3은 5가지 RPM 조건에서의 구성품 성능 값들이다.

Table 3 Performance parameter values of compressor and turbine

	RPM	50%	60%	70%	80%	90%
Compressor	PR	1.38111	1.59784	1.82956	2.14715	2.51732
	MFP	0.28153	0.43770	0.55305	0.66590	0.74716
	ETA	0.88376	0.87287	0.85212	0.80921	0.78778
Turbine	PR	1.23557	1.35727	1.50442	1.66064	1.79238
	MFP	0.40982	0.54162	0.59095	0.62968	0.62410
	ETA	0.55344	0.56185	0.59867	0.61593	0.63509

이렇게 구한 축척인자는 시스템 식별 기법을 통해 유사 엔진의 성능 선도를 스케일링 하는데 사용된다[27]. 사용된 마더맵으로는 압축기에는 Rad1kg.map를 터빈에는 NASAVG68.map를 사용하였다.

제 4 절 성능해석 결과

Figure 15는 시스템 식별 기법을 통해 스케일링 된 I-Jet 130 엔진의 성능 맵이다.

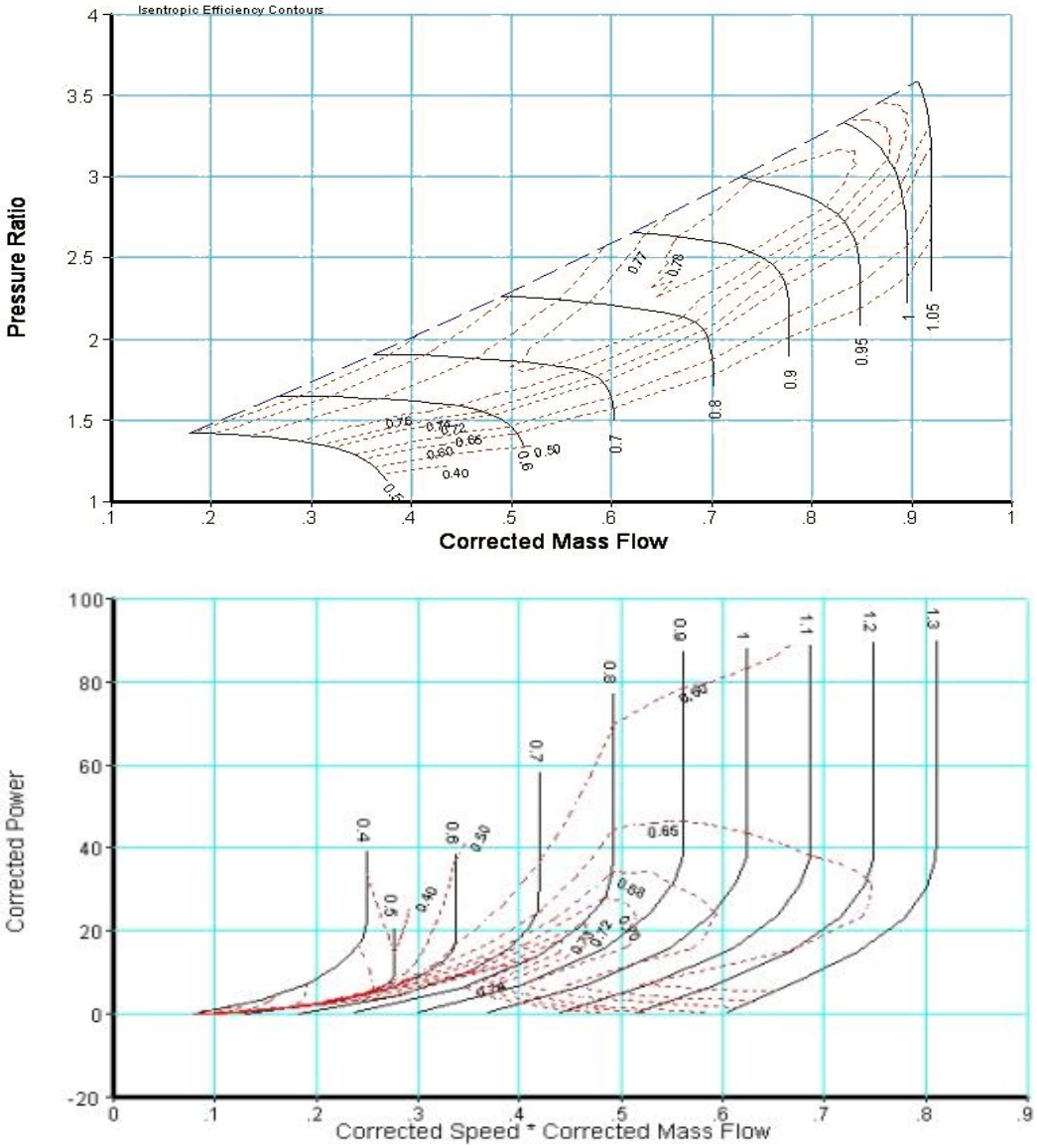


Fig. 15 Generated maps of compressor and turbine

C++를 이용하여 대상엔진의 정상상태 성능모델을 구성하였으며[28] 다시 70000~100000 RPM범위에서 가스터빈 시험 장치를 사용하여 얻어진 데이터로 실험 데이터를 통해 축척하는 방법으로 얻어진 성능선도를 사용한 성능 해석결과와 비교하여 검증하였다. Figure 16은 대상엔진의 성능모델의 Flow chart이며 Figure 17은 비교 검증의 결과를 나타낸 것이다.

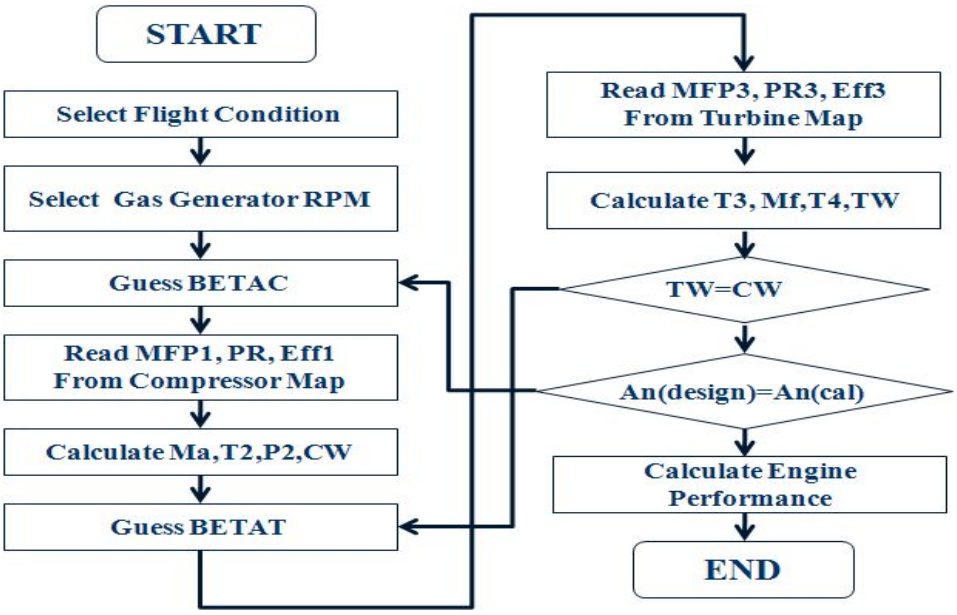


Fig. 16 Flow chart of performance simulation program using C++

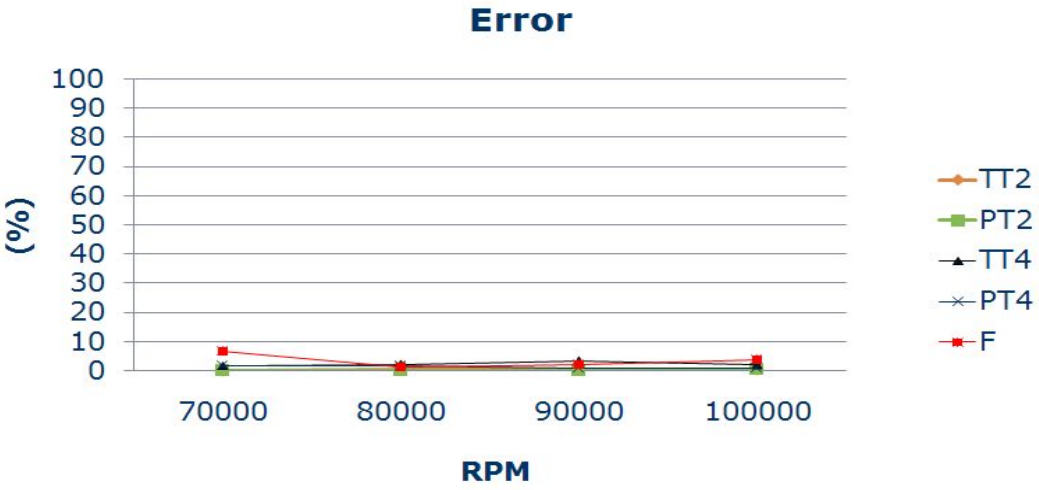


Fig. 17 Result of verification

TT2 : 압축기 출구 온도, PT2 : 압축기 출구 압력, TT4 : 터빈 출구 온도, PT4 : 터빈 출구 압력, F : 추력

해석 결과를 살펴보면, 해석 결과가 전체 작동영역에서 7% 이내의 오차를 보이는 것

을 볼 수 있다. 가장 높은 오차를 보인 70000 RPM은 50% RPM 정도 부근이며 고 RPM에서는 오차율이 적은 것을 확인 할 수 있었다.

이렇게 선행 연구된 시험 장치를 이용한 역모델링 연구를 통해 역식별 역모델링 기법에 대한 기초를 확립하고 또한 후에 실제 항공기용 터보프롭 엔진의 시험 데이터의 분석 및 활용에 대한 방안을 연구하였다.

제 3 장 터보프롭엔진의 역 모델링 연구

제 1 절 대상 엔진

연구대상 엔진으로는 P&W(Pratt & Whitney)사에서 개발한 PT6A-67 터보 프롭 엔진을 선정하였다. 이 엔진은 4단 축류와 1단 원심이 결합된 혼합형 압축기와 역 환형 기화식 연소기, 1단 압축기 터빈과 분리 축 방식의 정속 제어가 가능한 2단 축류 동력 터빈으로 구성되어 있다. 그 외 2단의 감속기어 박스가 있으며 1200hp로 출력이 Flat-rated 되어있다.

동력 터빈 축은 압축기 터빈 축과 별개로 감속기어 박스를 통해 후면 프로펠러를 회전시켜 추력을 얻는다. 이스라엘에서 운용중인 헤론이 PT6A-67 엔진을 사용하는 대표적인 무인기이다.

연구 대상 엔진의 지상정지 표준 대기상태에서의 설계점 주요 성능은 Table 4와 같다.

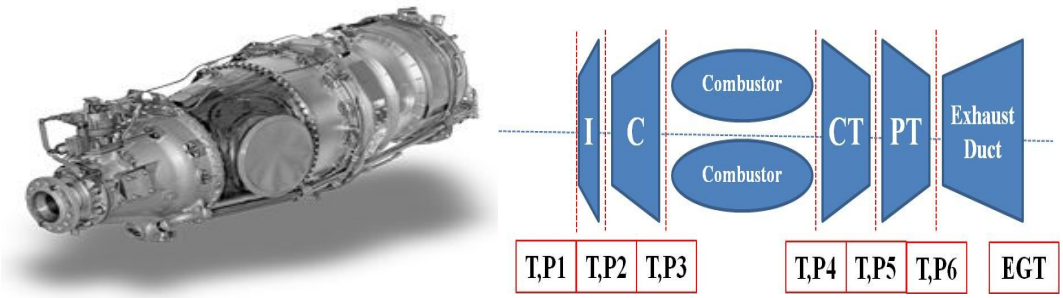


Fig. 18 PT6A-67 Turboprop Engine

Table 4 Design point performance of PT6A-67 turboprop

운 용 조 건	S/L, Static Standard
가스발생기 속도(rpm)	39,000
동력터빈 속도(rpm)	29,894
프로펠러 회전속도(rpm)	1,700
ITT(k)	1,113
출력 (HP)	1,700(1200으로 Flat-rated)
중량 (kg)	230

제 2 절 성능덱 데이터 분석

1. 입력조건

- 고도 조건 : 0 ~ 45000ft
- 속도 조건 : M0 ~ M0.5
- 엔진 출력 조건 : Max Takeoff(RC=1), Max Climb(RC=3), Max Cruise(RC=4), 80% Max Cruise(RC=5), 60% Max Cruise(RC=6), 30% Max Cruise(RC=7), Flight Idle(RC=8)
- Exhaust Area : 94 sq. inch
- Exhaust Angle : 6 도
- ESHP 계산을 위한 Propeller 효율 : 85%
- PRF(Probe Recovery Factor) : 1.0
- Wf Option : Average New Engine with LHV 18,400 BTU/lb
- Charge Heating : 3℃
- Ram recovery : 0.9
- Accessory reduction gearbox power HP: 12.1 HP(9KWatt)
- Interstage Bleed : 0
- Compressor Del. Bleed : 0.00
- Intake Pressure Loss : 0.0
- Exhaust Pressure Loss : 0.0

- Exhaust Depression : 0.0
- Propeller Shaft Speed : 1700. constant
- Rating Code : 1, 3, 4, 5, 6, 7, 8

2. 성능덱 데이터 분석

PT6A-67 엔진의 입력값에는 고도, 속도, 외기온도조건, 엔진 조건(rating code; max TO, Max Climb, Max Cruise, 80% Max CR, 60% Max CR, 30% Max CR, Flight idle), 압축기 블리드공기 분출, 동력분출(Power Takeoff), 배기구 면적 및 각도, 프로펠러 효율, Inlet charge heating, ram recovery, propeller shaft speed가 있고 출력값으로는 SHP, Fuel flow, sfc, gasgenerator speed, air flow, P1, P2.5, P3, P7, P2.5 bleed, P3 bleed, T1, T2.5, T3, T7, ITT이 있다[29].

엔진 구성품 모델링이나 엔진 성능 모사를 위해서는 성능덱으로 얻을 수 있는 출력값만으로는 데이터가 부족한데 이것을 해결하기 위해 열역학적 관계식과 같은 계열의 유사엔진에 대한 스케일링을 이용하여 부족한 데이터를 얻어냈다. Table 5는 PT6A-67 엔진의 M200 형상 지상 Take-off 조건의 주요 성능값과 같은 PT6A계열인 PT6A-62 엔진의 지상 설계점의 성능값이다.

Table 5 Design point performance of PT6A-67 and PT6A-62

	PT6A-67	**PT6A-62**
공기유량(kg/s)	4.613	4.18
연료유량(kg/s)	0.087643	0.08178
압축기 압력비	9.26	9.4
압축기 출구온도(k)	615.33	616.67
압축기 터빈 입구온도(k)	1288.2	1292.62
축마력(SHP)	1200	1150
비연료소모율(kg/kw • hr)	0.35259	0.3432
가스발생기 회전수	36308	37468
프로펠러 회전수(RPM)	1700	2000

유사엔진의 자료를 이용하여 터빈의 각 조건별 추가적으로 계산해 낸 성능 값은 다음과 같다.

- 수정 RPM
- 압축기 터빈 입구온도 T4
- 압축기 터빈 입구압력 P4
- 동력 터빈 입구압력 P5
- 배기 제트 속도

터빈 압력비의 경우 터빈의 효율의 함수이기 때문에 열역학적 관계식만으로는 계산할 수 없기 때문에 Table 5와 같이 유사한 성능을 가진 PT6A-62 엔진의 터빈 효율의 성능 값을 참고하여 계산을 수행하였다. 계산에는 시스템 식별기법이 사용되었다[30].

제 3 절 구성품 성능선도 역생성

1. 압축기

압축기 구성품의 성능 선도는 온도와 압력에 관한 함수이기 때문에 다양한 조건에서의 성능덱 데이터를 이용하여 압축기의 성능값을 계산한 후 온도와 압력에 따라 정리하였다.

압축기 구성품의 유량함수 MFP1은 공기유량을 대기온도, 압력을 이용하여 무차원화시키는 것으로서 식 (9)으로부터 구할 수 있고, 압축비 PR1은 대기 압력과 압축기 출구 압력의 압력비로서 식 (10)으로 구할 수 있다. 마지막으로 압축기 효율 는 식 (11)로부터 구한다.

$$MFP1 = Ma\frac{\sqrt{T1}}{P1} \tag{9}$$

$$PR1 = \frac{P3}{P1} \tag{10}$$

$$T3 - T1 = \frac{T1}{\eta}\left[PR1^{\frac{1-\gamma}{\gamma}} - 1\right] \tag{11}$$

이렇게 계산된 압축기 성능값은 온도와 압력에 따라 새로 정리하였다. Figure 19는 압축기 맵 생성의 Flow Chart이다.

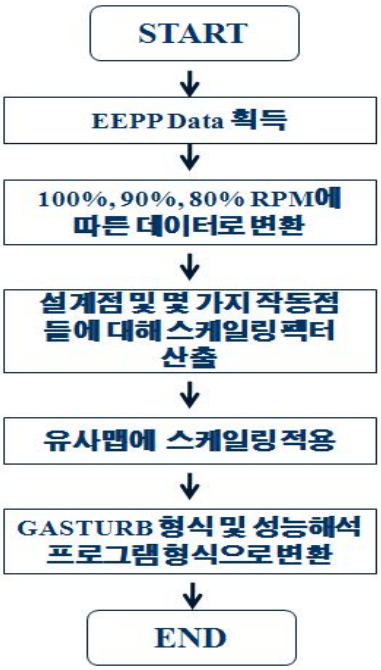


Fig. 19 Flow of Compressor Map Generation

먼저 압축기의 성능값인 MFP1, PR1, 을 계산한 뒤, 온도와 압력에 따라 정리한다. 기본적으로 성능덱 데이터는 RC(Rating Code, 비행 상태에 따른 엔진 출력)값으로 정리되어있는데 이를 다시 RPM값에 따른 데이터로 변환한다. 가스제너레이터 RPM은 먼저 온도와 압력에 따른 수정RPM으로 변환한 후 RC 1(Take-off 상태, 100% RPM)상태의 RPM에 대한 %RPM으로 다시 변환 시킨다. 보간법과 보외법을 사용하면 GASTURB 맵 기준으로 80%, 90%, 100% RPM 일 때의 성능 데이터 값으로 변환 할 수 있다.

이렇게 구해진 3가지 %RPM 일 때의 성능 데이터를 기준으로 지상 설계점과 몇 가지 작동점들을 뽑아 유사 GASTURB 성능 선도(Ruston 6MW)에 식 (12), (13), (14)를 이용하여 스케일링을 수행한다. 80%, 90%, 100% 외의 부분부하 상태에 대한 스케일링은 100%일 때의 스케일링 팩터를 이용하여 스케일링을 수행하였다[31].

$$PR = \frac{PR_{design} - 1}{PR_{Map\,design} - 1}(PR_{Map} - 1) + 1 \tag{12}$$

$$MFP = \frac{MFP_{design}}{MFP_{Mapdesign}} \times MFP_{Map} \tag{13}$$

$$\eta = \frac{\eta_{design}}{\eta_{Mapdesign}} \times \eta_{Map} \tag{14}$$

Figure 20은 역식별 기법(Inverse Identification Method)으로 생성된 PT6A-67의 압축기 성능선도이다.

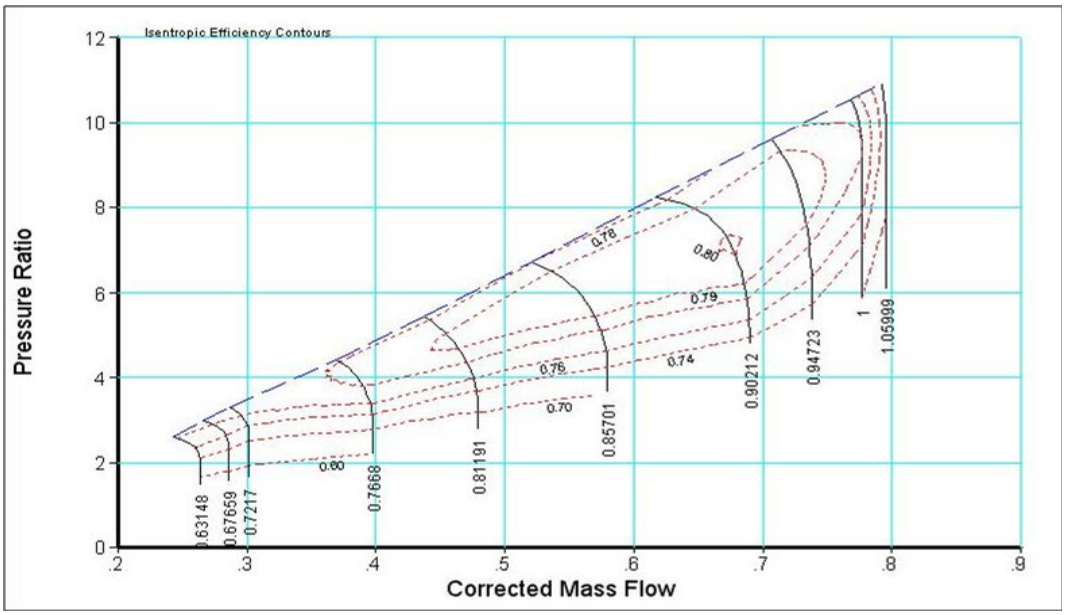


Fig. 20 Generated Compressor Map of PT6A-67 Using Inverse Identification Method

2. 가스제너레이터 터빈

가스제너레이터 터빈 구성품의 성능 선도 역시 온도와 압력에 관한 함수이기 때문에 다양한 조건에서의 성능덱 데이터를 이용하여 압축기 터빈의 성능값을 계산한 후 온도와 압력에 따라 정리하였다. 하지만 터빈의 경우 압축기와 달리 성능덱 데이터 중 압력 데이터가 부족하기 때문에 유사 엔진인 PT6A-62 엔진의 가스제너레이터 터빈 성능선도로부터 참고하여 계산을 수행 하였다. 가스제너레이터 터빈 성능선도 역시 압축기와 같은 절차에 따라 생성하였다. Figure 21은 역식별 기법(Inverse Identification Method)으로 생성된 PT6A-67의 가스제너레이터 터빈 성능선도이다.

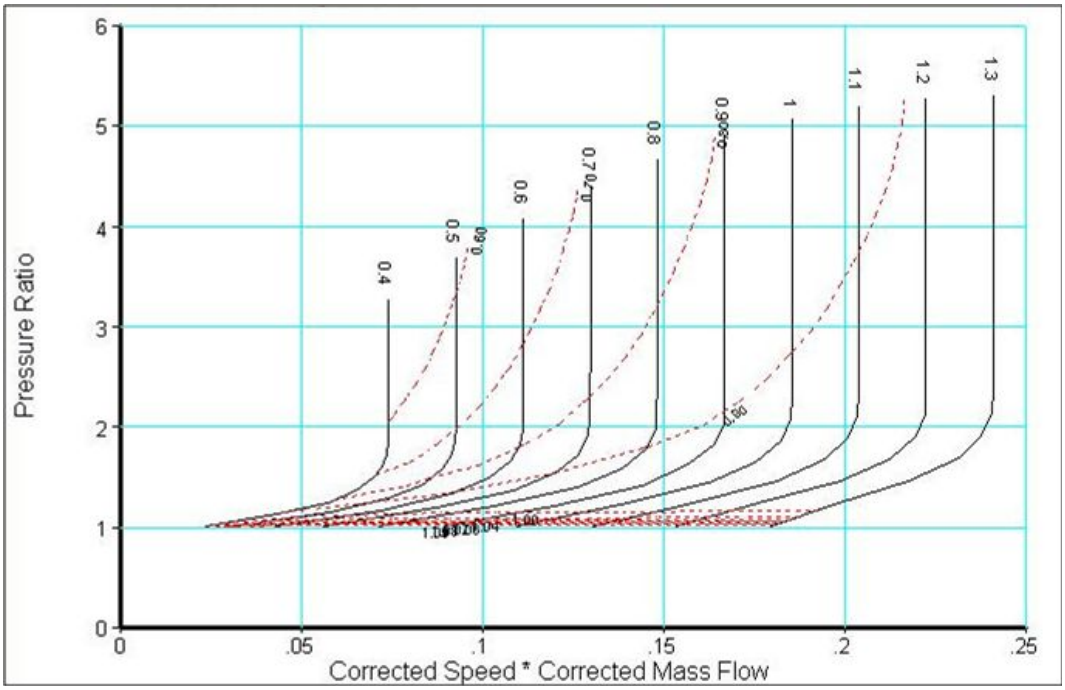


Fig. 21 Generated HP Turbine Map of PT6A-67 Using Inverse Identification Method

3. 동력 터빈

동력 터빈 구성품의 성능 선도 역시 온도와 압력에 관한 함수이기 때문에 다양한 조건에서의 성능덱 데이터를 이용하여 동력 터빈의 성능값을 계산한 후 온도와 압력에 따라 정리하였다. 동력 터빈의 경우도 가스제너레이터 터빈과 마찬가지로 성능덱 데이터 중 압력 데이터가 부족하기 때문에 유사 엔진인 PT6A-62 엔진의 동력 터빈 성능선도로부터 참고하여 계산을 수행 하였다. 동력 터빈 성능선도 역시 압축기와 같은 절차에 따라 생성하였다. Figure 22는 역식별 기법(Inverse Identification Method)으로 생성된 PT6A-67의 동력 터빈 성능선도이다.

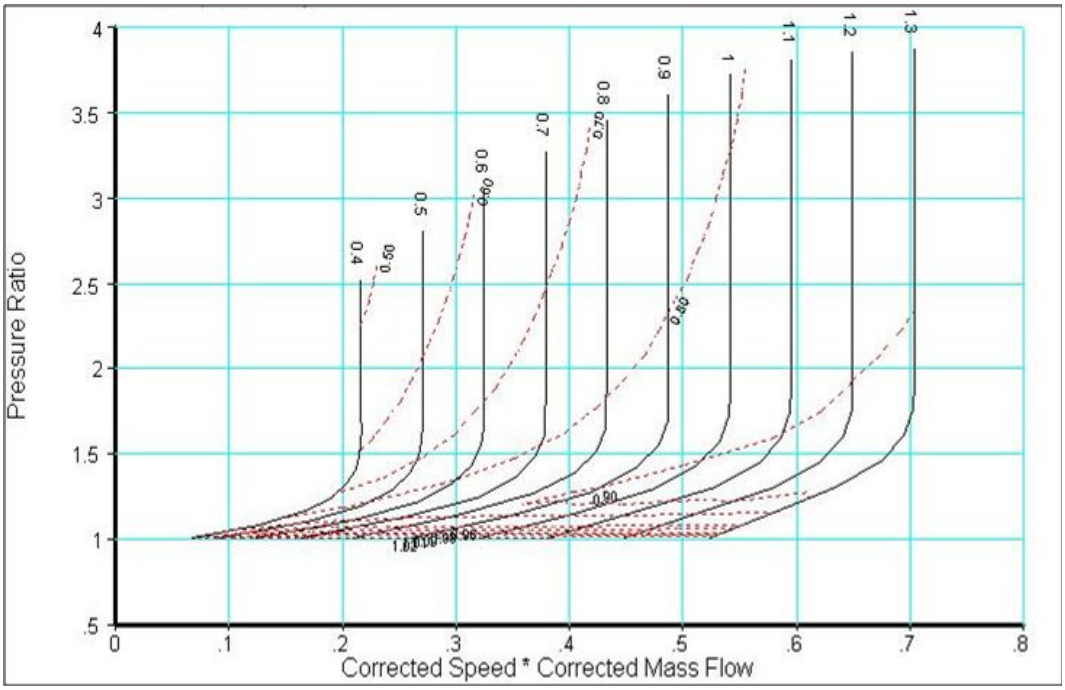


Fig. 22 Generated LP Turbine Map of PT6A-67 Using Inverse Identification Method

제 4 절 엔진 성능 모델링 및 생성된 구성품 성능선도 검증

개선된 구성품 성능선도의 검증을 위해 엔진의 각 구성품인 흡입구, 압축기, 연소기, 압축기 터빈, 동력터빈 등은 C++를 이용하여 모델링을 수행하였다. Figure 23은 엔진 성능 모델의 알고리즘이다.

다양한 조건에서의 성능덱 데이터를 이용하여 생성한 구성품들의 성능 선도기 때문에 구성품들은 하나의 Beta Line(작동점)을 따라 계산된다. 즉 성능선도는 하나의 작동점에 따라 유량조합이 이루어진 세트로 이루어져 있다. 먼저 비행 상태로서 고도, 온도변화, 속도, RC를 입력값으로 기본적으로 흡입구로 들어오는 대기 상태(온도, 압력)가 계산되고, 흡입구 모델을 지나 구성품 들의 Beta Line을 가정한 후 압축기 성능 선도로부터 성능 값을 읽어 들여 압축기 모델에 따라 압축기 출구 온도와 압력, 공기유량과 출력이 결정된다. 그 다음 연소기 모델을 지나 다시 가정된 Beta Line의 압축기 터빈 성능 선도로부터 성능 값을 읽어 들인 후 압축기 터빈 입구온도, 압력, 연료유량, 출구 온도, 압

축기 터빈 출력 등이 계산된다. 그 후 압축기 출력과 압축기 터빈 출력이 같아질 때까지 Beta Line이 재 가정되어 반복 계산 후, 확실한 작동점이 찾아지면 다시 동력 터빈의 성능 선도로부터 성능 값을 읽어 들여 동력터빈 출구 온도, 압력, 출력 등 엔진의 성능을 계산한다.

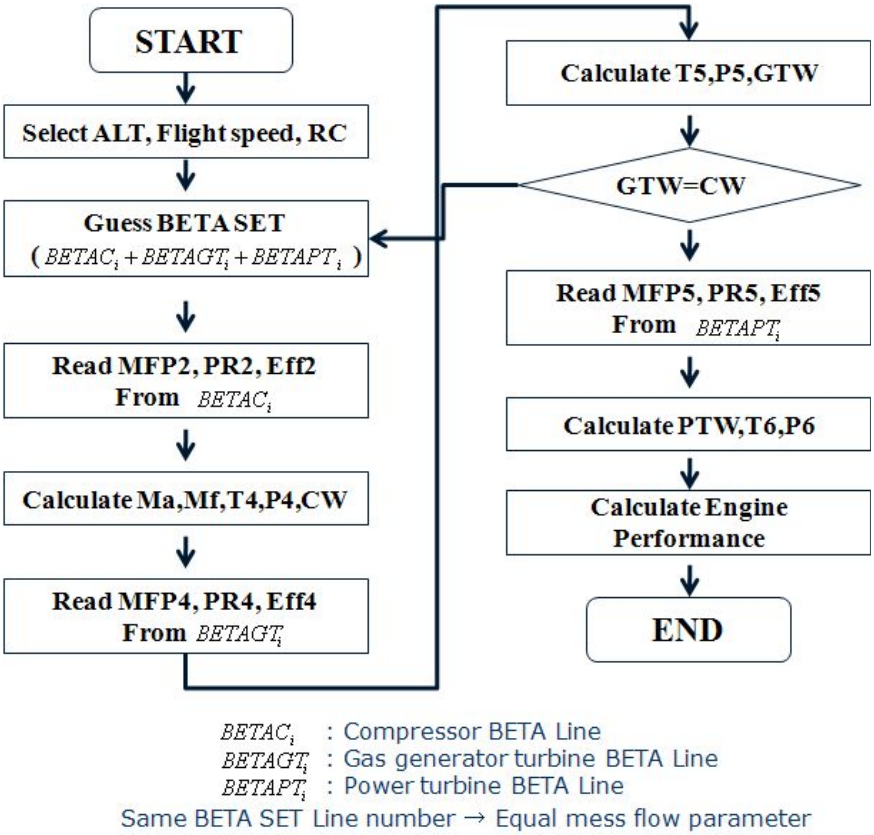


Fig. 23 Flowchart of base performance simulation program

개선된 구성품 성능선도와 모델 프로그램의 검증을 위해 구성품 성능선도를 포함한 엔진 성능모델의 성능해석 결과와 성능덱 성능 데이터의 비교하였다. 해석결과는 압축기 출구 온도, 압축기 터빈 입구 온도, 공기유량, 연료유량, 압축기 출력, 동력터빈 출력, SFC 이다. 해석은 고도와 RC에 따라 이루어 졌다. 앞서 정의한 각 RC를 설명의 편의를 위해 다음과 같이 다시 언급한다.

- Max Take-off(RC=1), Max Climb(RC=3), Max Cruise(RC=4), 80% Max

Cruise(RC=5), 60% Max Cruise(RC=6), 30% Max Cruise(RC=7)

1. RC-1

엔진 출력조건 RC 1은 비행 상태가 Take-off를 상정한 엔진 출력 상태이며, 이때의 가스제너레이터 RPM은 100%이다. Figure 24, 25는 RC 1일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

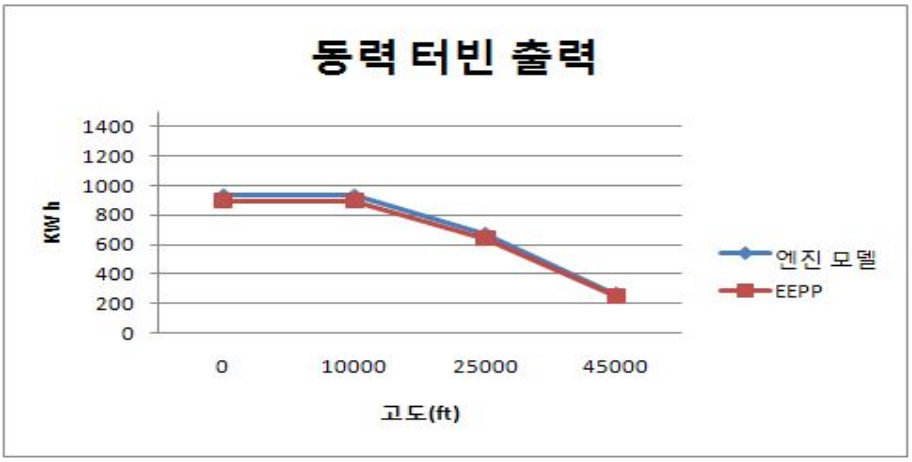


Fig. 24 Power Turbine Work

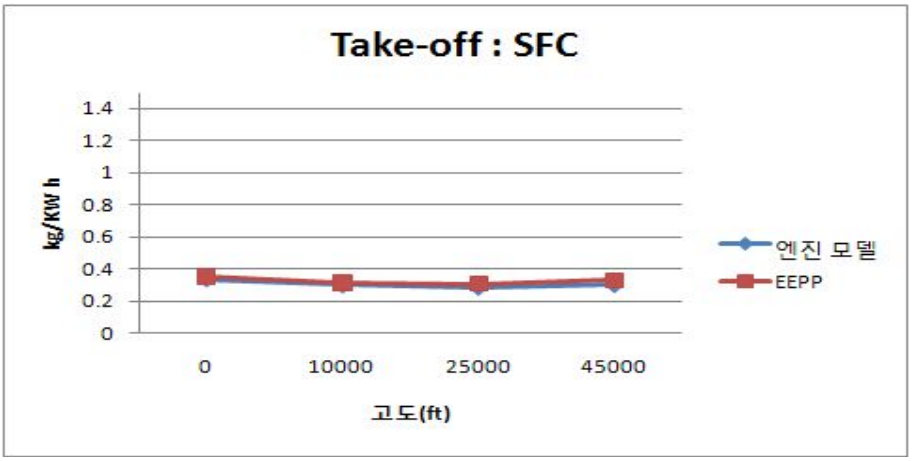


Fig. 25 SFC

2. RC-3

엔진 출력조건 RC 3은 비행 상태가 Climb를 상정한 엔진 출력 상태이며, 이때의 가스제너레이터 RPM은 97~99%이다. Figure 26, 27은 RC 3일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

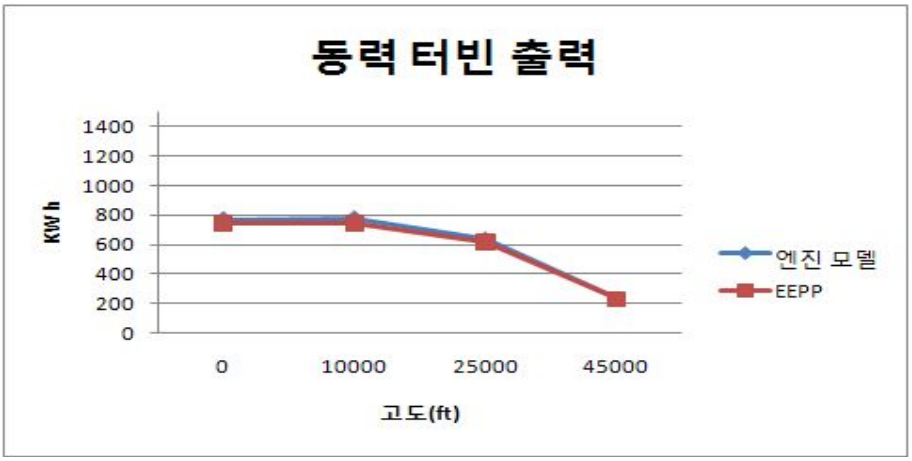


Fig. 26 Power Turbine Work

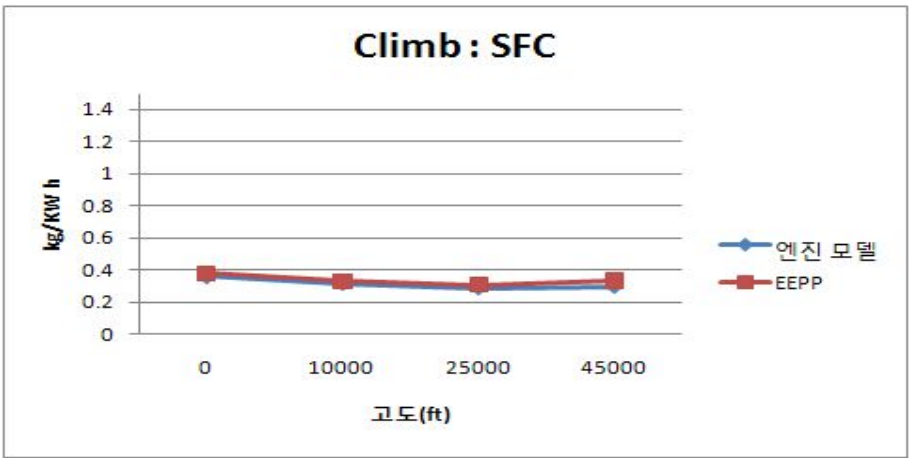


Fig. 27 SFC

3. RC-4

엔진 출력조건 RC 4은 비행 상태가 Max Cruise를 상정한 엔진 출력 상태이며, 이때의 가스 제너레이터 RPM은 94~97%이다. 다음 Figure 28, 29는 RC 4일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

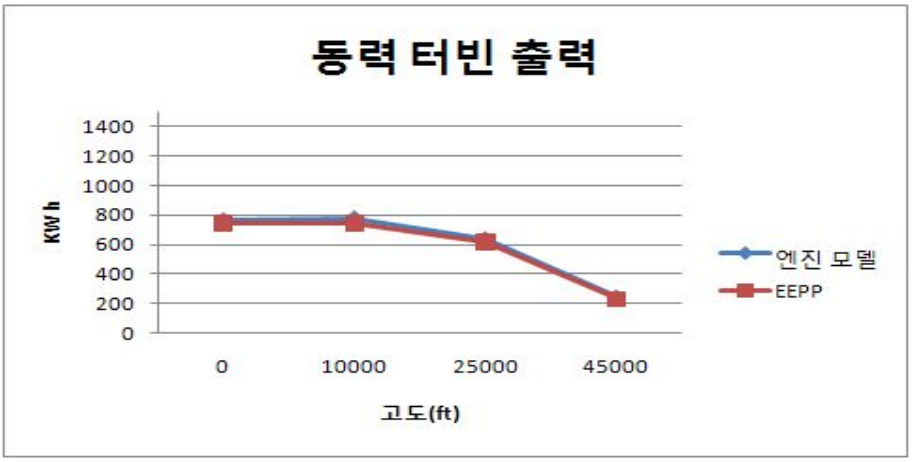


Fig. 28 Power Turbine Work

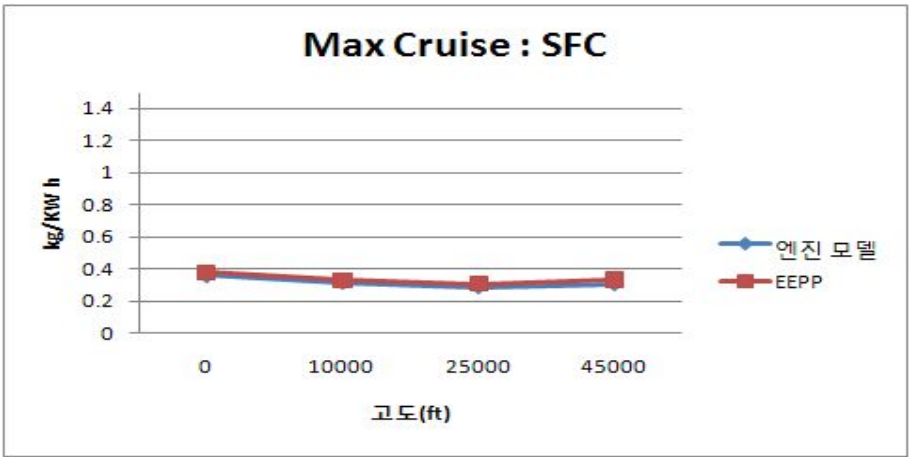


Fig. 29 SFC

4. RC-5

엔진 출력조건 RC 5는 비행 상태가 80% Cruise를 상정한 엔진 출력 상태이며, 이때의 가스 제너레이터 RPM은 91~94%이다. Figure 30, 31은 RC 5일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

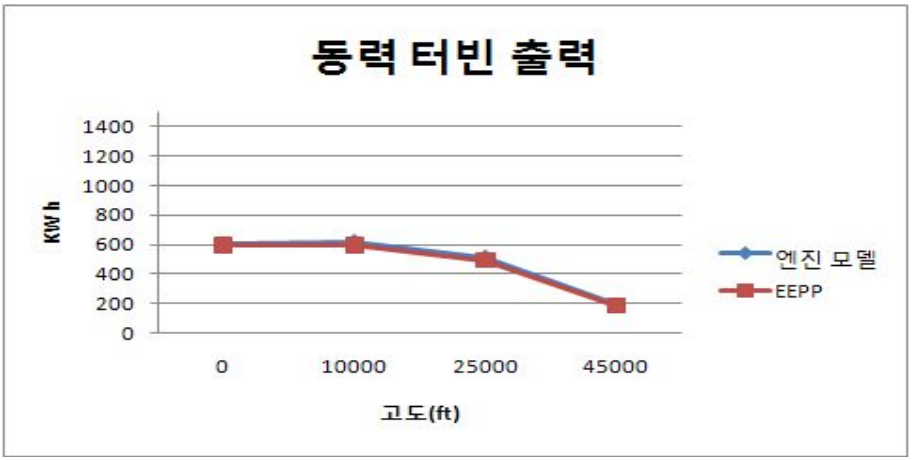


Fig. 30 Power Turbine Work

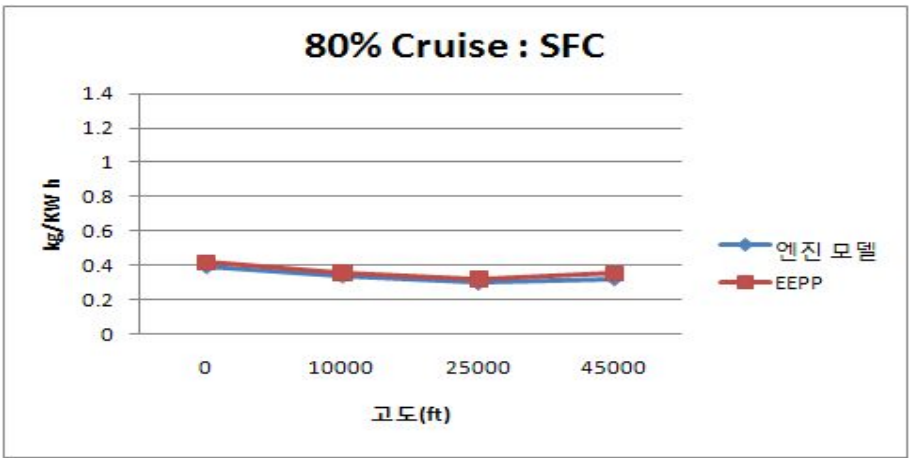


Fig. 31 SFC

5. RC-6

엔진 출력조건 RC 6는 비행 상태가 60% Cruise를 상정한 엔진 출력 상태이며, 이때의 가스 제너레이터 RPM은 87~91%이다. Figure 32, 33는 RC 6일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

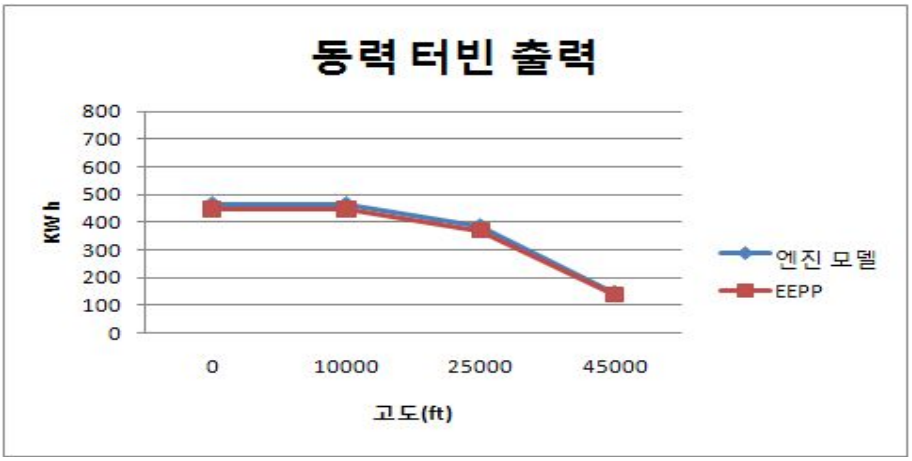


Fig. 32 Power Turbine Work

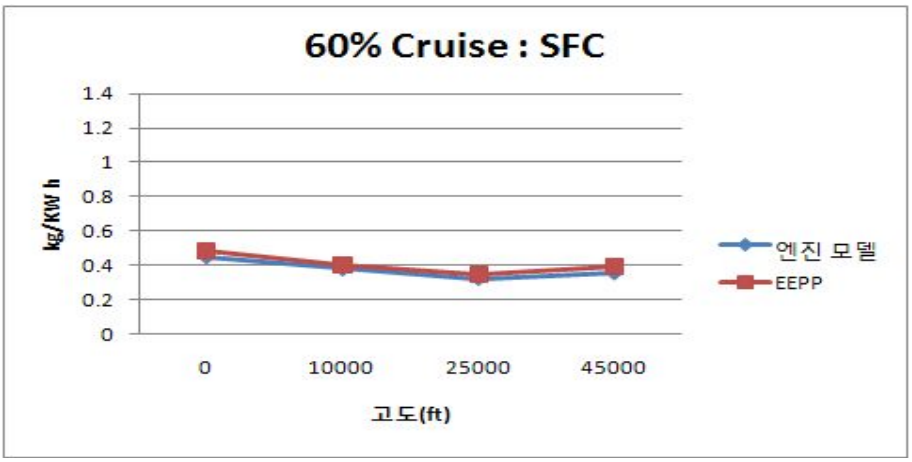


Fig. 33 SFC

6. RC-7

엔진 출력조건 RC 7은 비행 상태가 30% Cruise를 상정한 엔진 출력 상태이며, 이때의 가스 제너레이터 RPM은 81~87%이다. Figure 34, 35는 RC 7일 때 고도에 따른 엔진 모델과 성능덱 성능 해석 결과이다.

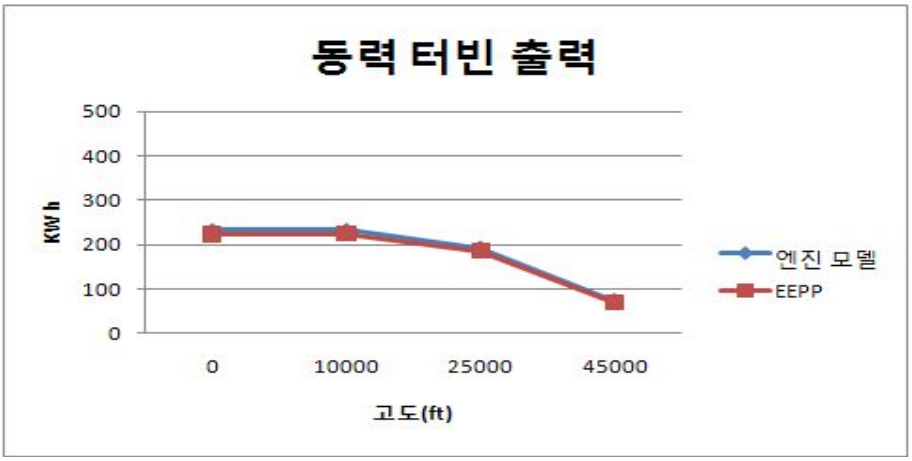


Fig. 34 Power Turbine Work

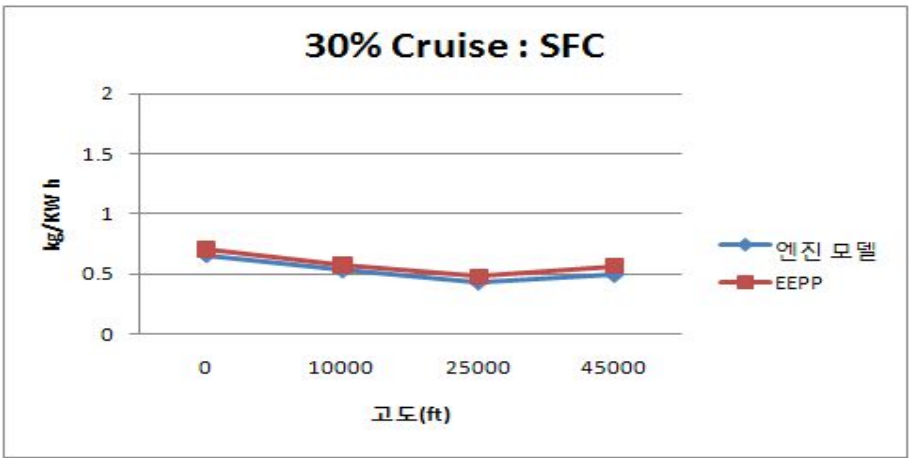


Fig. 35 SFC

성능 해석 결과를 보면 전체적으로 1~3%의 오차를 보이는 것을 알 수 있었다. 그러나 전반적으로 구성품생성시 제시한 역 모델기법과 이를 이용한 엔진성능모델이 성능덱과 비교를 통해 타당한 방법임을 검증되었다고 볼 수 있다.

제 4 장 터보프롭엔진의 인공지능 진단 연구

제 1 절 주요 구성품 손상에 따른 엔진 성능저하 특성 연구

압축기, 연소기, 터빈의 주요 손상원인은 외부물질의 흡입과 연소산화물의 흡착에 따른 오염, 침식, 부식 등이 있다.

오염이 발생하면 유로가 협소해져 통과하는 유량함수가 감소하고 그로인해 효율이 저하된다. 이는 출력의 감소와 연료소모율의 증가를 가져오게 된다.

침식은 유도각의 변화를 유발하여 흐름 박리와 스톨을 발생시키며 부분부하에서의 서지 발생 가능성을 증가시킨다. 이는 구성품의 등엔트로피 효율을 감소시키는 원인이 되며 유량함수 증가를 가져온다[32].

부식은 부품의 피로강도를 감소시키며 등엔트로피 효율 감소의 원인이 된다.

본 연구에서는 터보프롭 엔진에서 주로 발생되는 압축기 오염과 압축기터빈 및 동력터빈 침식의 상황을 가상하고 그때의 주요 구성품 계측변수 변화를 분석하였다.

1. 성능저하 원인에 따른 계측변수 분석

엔진 손상은 압축기와 터빈에서 각각 손상이 발생되는 경우, 두 구성품에서 동시에 손상이 발생되는 경우, 모든 구성품에서 동시에 손상이 발생되는 경우 등 다양한 상황을 가정하였다. 가정된 각 Case의 상태진단 변수의 변화량은 Table 6, 7과 같다.

Table 6은 각 구성품에서 단일 손상이 발생되는 경우로 Case 1 ~ Case 9는 압축기에서 오염이 발생하여 유량함수와 효율이 1%에서 5% 범위에서 감소하는 경우, Case 10 ~ Case 14는 압축기 터빈에서 침식이 발생하여 유량함수는 증가하고 효율은 감소하는 경우, Case 15 ~ Case 19는 동력터빈에서 침식이 발생하여 유량함수의 증가와 효율 감소가 나타나는 경우이다.

Table 7은 각 구성품에서 다중 손상이 발생되는 경우로 Case 20 ~ Case 28은 두 구성품에서 동시에 손상이 발생한 경우이며 Case 29 ~ Case 35는 압축기, 압축기터빈, 동력터빈 모두에서 동시에 손상이 발생한 경우이다.

Table 6 Supposed performance fallen rate(Single fault)

CASE		압축기		압축기터빈		동력터빈	
		유량	효율	유량	효율	유량	효율
압축기 오염	1	-1	-1	0	0	0	0
	2	-2	-2	0	0	0	0
	3	-3	-3	0	0	0	0
	4	-4	-4	0	0	0	0
	5	-5	-5	0	0	0	0
	6	-1	-5	0	0	0	0
	7	-2	-4	0	0	0	0
	8	-4	-2	0	0	0	0
	9	-5	-1	0	0	0	0
압축기터빈 침식	10	0	0	+1	-1	0	0
	11	0	0	+2	-2	0	0
	12	0	0	+3	-3	0	0
	13	0	0	+4	-4	0	0
	14	0	0	+5	-5	0	0
동력터빈 침식	15	0	0	0	0	+1	-1
	16	0	0	0	0	+2	-2
	17	0	0	0	0	+3	-3
	18	0	0	0	0	+4	-4
	19	0	0	0	0	+5	-5

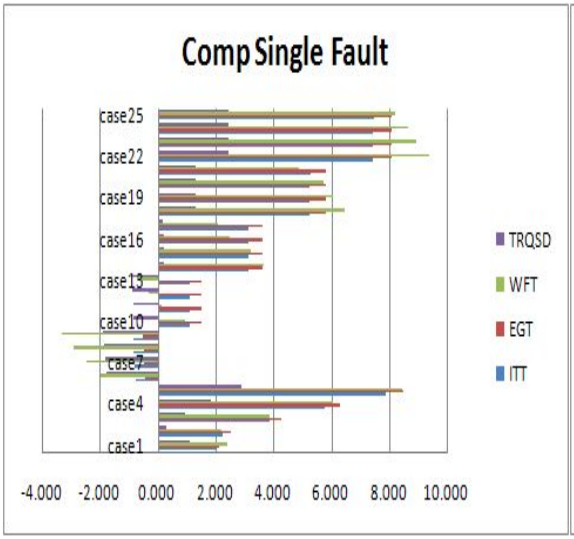
Table 7 Supposed performance fallen rate(Multiple faults)

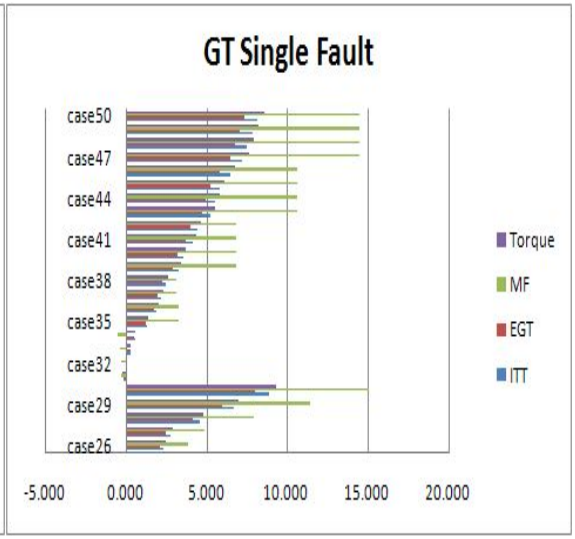
CASE		압축기		압축기터빈		동력터빈	
		유량	효율	유량	효율	유량	효율
압축기 오염 &압축기 터빈 침식	20	-3	-3	+3	-3	0	0
	21	-1	-3	+3	-1	0	0
	22	-3	-1	+1	-3	0	0
압축기 오염 &동력 터빈 침식	23	-3	-3	0	0	+3	-3
	24	-1	-3	0	0	+3	-1
	25	-3	-1	0	0	+1	-3
압축기 터빈 &동력 터빈 침식	26	0	0	+3	-3	+3	-3
	27	0	0	+1	-3	+3	-1
	28	0	0	+3	-1	+1	-3
압축기 오염 &압축기 터빈 침식 &동력 터빈 침식	29	-3	-3	+3	-3	+3	-3
	30	-1	-3	+3	-1	+1	-3
	31	-3	-1	+1	-3	+3	-1
	32	-1	-3	+1	-3	+3	-1
	33	-3	-1	+3	-1	+1	-3
	34	-3	-1	+3	-1	+3	-1
	35	-1	-3	+1	-3	+1	-3

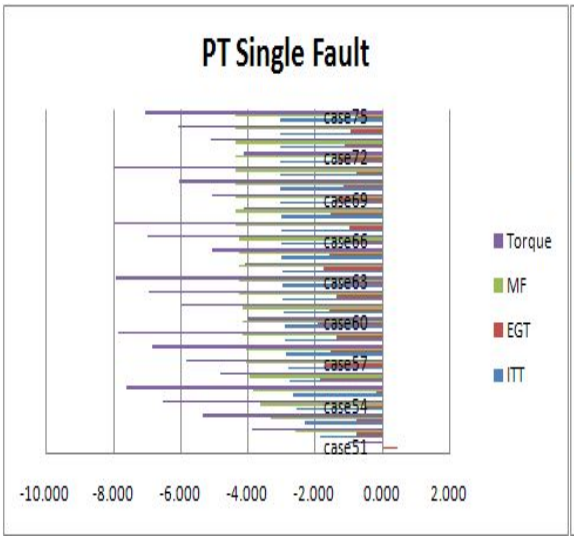
Table 8 Supposed performance fallen rate

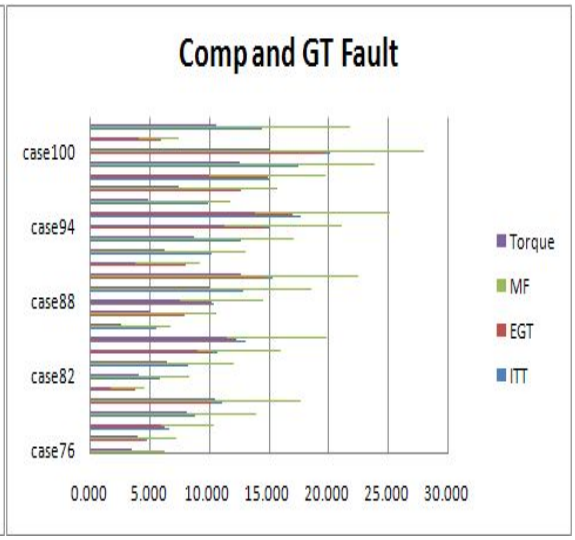
Compressor efficiency	-1~-5%
Compressor mass flow	-1~-5%
High pressure turbine efficiency	-1~-5%
High pressure turbine mass flow	+1~+5
Power turbine efficiency	-1~-5%
Power turbine mass flow	+1~+5

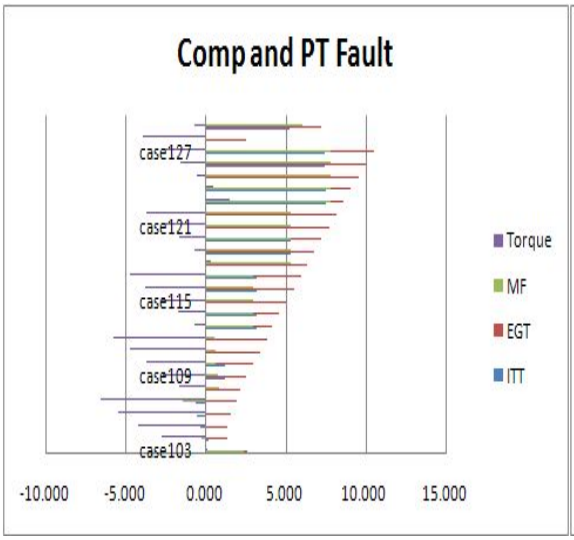
Table 8과 같이 한 조건에서의 손상 Case는 283개, 고도, 속도, RC에 따라 17가지 조건으로 총 4811 Case에 대한 성능 데이터베이스를 구축하였다. 시뮬레이션에는 C++로 제작된 기준엔진 모델을 사용하였다.

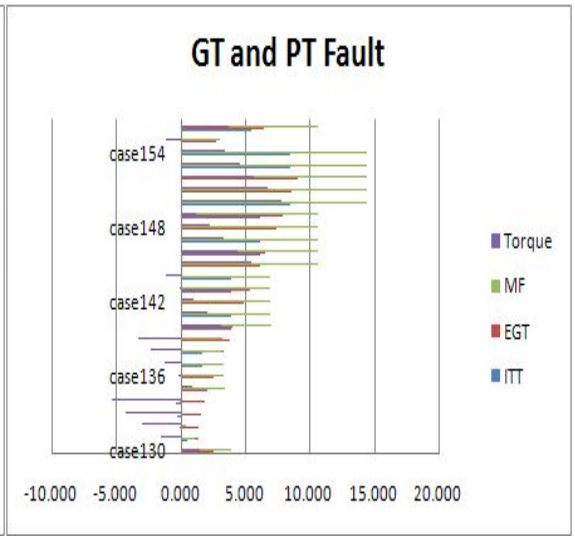

Comp Single Fault
case25
case22
case19
case16
case13
case10
case7
case4
case1
-4.000
-2.000
0.000
2.000
4.000
6.000
8.000
10.000
TRQSD
WFT
EGT
ITT


GT Single Fault
case50
case47
case44
case41
case38
case35
case32
case29
case26
-5.000
0.000
5.000
10.000
15.000
20.000
Torque
MF
EGT
ITT


PT Single Fault
case75
case72
case69
case66
case63
case60
case57
case54
case51
-10.000
-8.000
-6.000
-4.000
-2.000
0.000
2.000
Torque
MF
EGT
ITT


Comp and GT Fault
case100
case94
case88
case82
case76
0.000
5.000
10.000
15.000
20.000
25.000
30.000
Torque
MF
EGT
ITT


Comp and PT Fault
case127
case121
case115
case109
case103
-10.000
-5.000
0.000
5.000
10.000
15.000
Torque
MF
EGT
ITT


GT and PT Fault
case154
case148
case142
case136
case130
-10.000
-5.000
0.000
5.000
10.000
15.000
20.000
Torque
MF
EGT
ITT

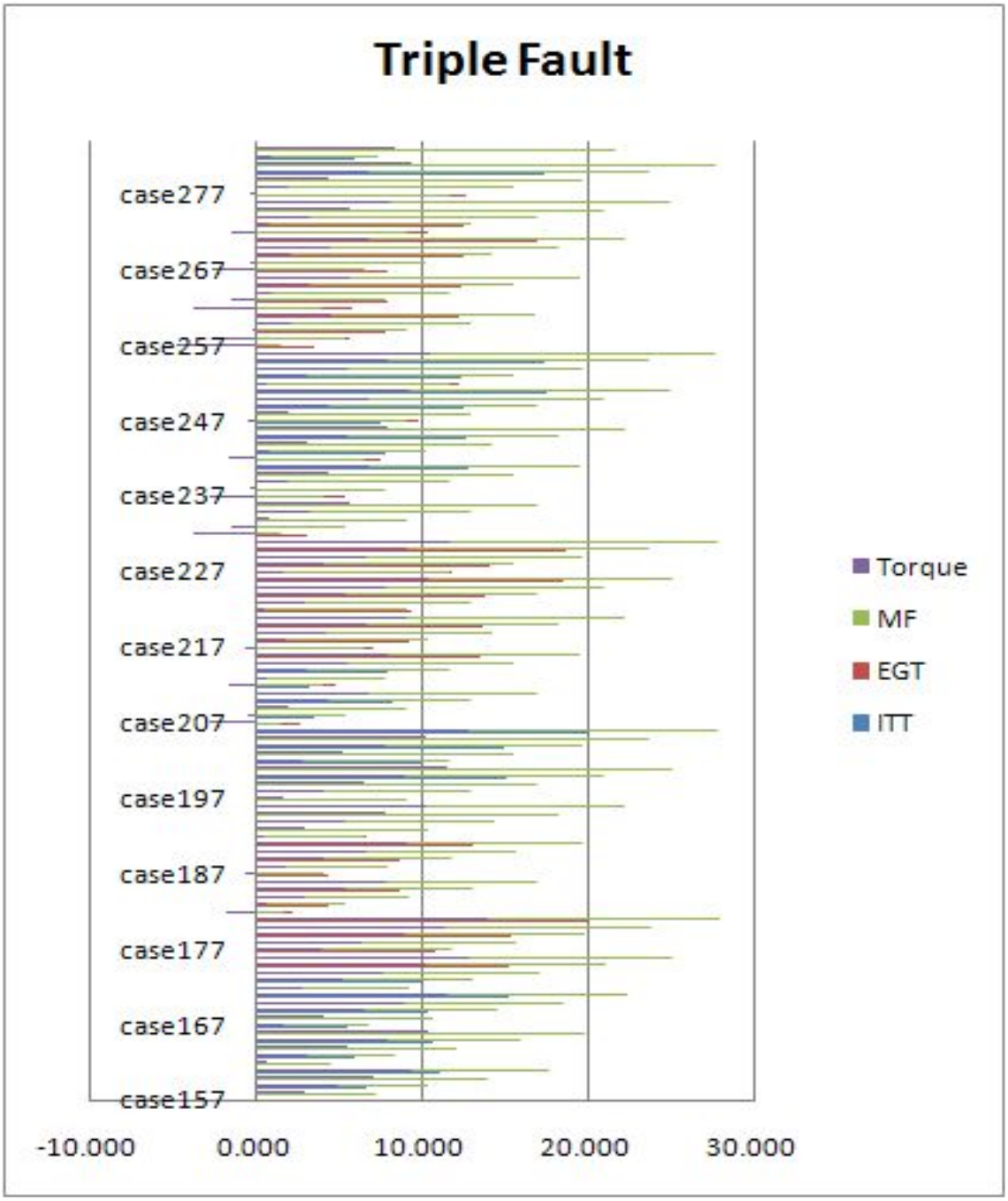


Fig. 36 Performance fallen rate of 45000ft, M0.3, RC4 Condition

Figure 36은 17가지 조건 중 한가지인 45000ft, M0.3, RC4일 때 손상에 따른 성능저하이다.

제 2 절 퍼지 및 신경회로망을 이용한 다중 손상 진단

최근에는 이러한 Model-based 진단 기법의 성능향상을 위해 신경망 이론, 퍼지 논리, 유전알고리즘과 같은 인공지능 이론을 적용한 진단 전문가 시스템이 연구되고 있다. 그 중 신경회로망은 우수한 학습 성능으로 인해 진단시스템에 가장 많이 이용되고 있으나 학습할 데이터가 많아지면 훈련하는데 시간이 오래 걸리고 정확성이 떨어질 수 있는 단점이 있다[33]. 또한 각 구성품의 단일 손상에서 여러 구성품에서 발생하는 다중 손상 탐지를 효과적으로 하기 위해서는 그 구조가 매우 복잡해진다.

따라서 본 연구에서는 1차적으로 손상된 구성품을 식별하고 그에 맞게 훈련된 신경회로망을 적용하여 정량적으로 엔진 상태를 진단하는 알고리즘을 사용하였다. 손상된 구성품을식별하는 알고리즘에는 퍼지로직을 이용하였으며 손상된 구성품의 계측데이터 학습에는 FFBP(Feed Forward Back Propagation) 알고리즘을 이용하였다. 다중손상 진단을 위한 계산 순서는 Figure 37과 같다.

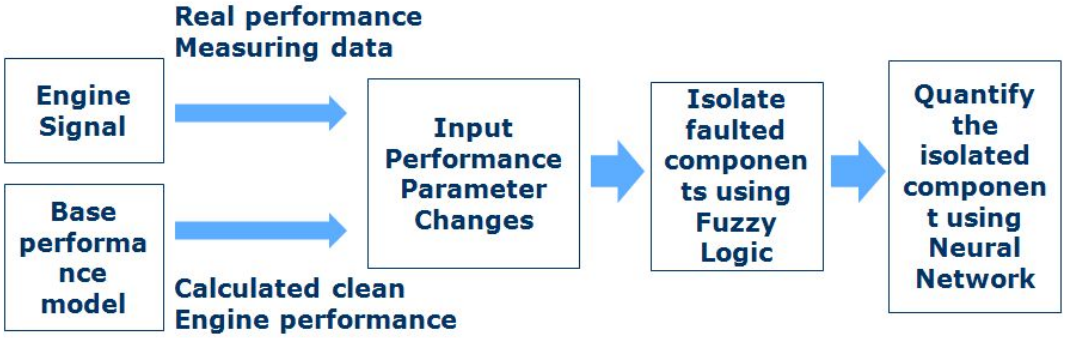


Fig. 37 Flowchart of proposed diagnostic system

1. 퍼지로직을 이용한 손상된 구성품 식별

구성품의 손상 정도를 정량적으로 진단하기 전에 엔진의 손상 패턴을 식별하기 위한 퍼지 알고리즘을 제안하였다. 추진시스템의 계측변수는 동력터빈 입구온도, 배기가스 온도, 연료유량, 토크이다. 주요 구성품의 손상 패턴은 압축기, 압축기터빈, 동력터빈에서 각각 손상이 발생하는 경우와 2개 구성품에서 동시에 손상이 발생하는 경우, 그리고 세

구성품에서 동시에 손상이 발생하는 경우 등을 고려해볼 수 있다.

본 연구에서는 주요구성품에서 발생할 수 있는 손상패턴을 Table 9와 같이 7가지 경우로 분류하였다.

Table 9 Considered component fault pattern cases

FP 1	Compressor fouling
FP 2	Compressor turbine erosion
FP 3	Power Turbine Erosion
FP 4	Comp. Fouling & Comp. turbine erosion
FP 5	Comp. Fouling & & Power Turbine Erosion
FP 6	Comp. turbine erosion & Power Turbine Erosion
FP 7	Comp. Fouling & Comp. turbine erosion & power turbine erosion

각각의 손상 패턴에 따라 계측파라미터의 변화를 살펴보면 Table 10과 같은 경향을 갖는다.

Table 10 Measured parameter change (MPC) trends depending on component fault patterns

	ITT	EGT	WF	Torque
FP 1	+	+	++	+
FP 2	+	+	+++	+
FP 3	-	+	-	--
FP 4	++	++	+++	++
FP 5	++	++	++	+
FP 6	++	++	+++	+
FP 7	+++	+++	+++	+

Figure 38은 손상된 구성품의 식별을 위한 mamdani-type 퍼지추론 시스템이며 MATLAB의 FIS 에디터를 이용하여 제작하였다[34].

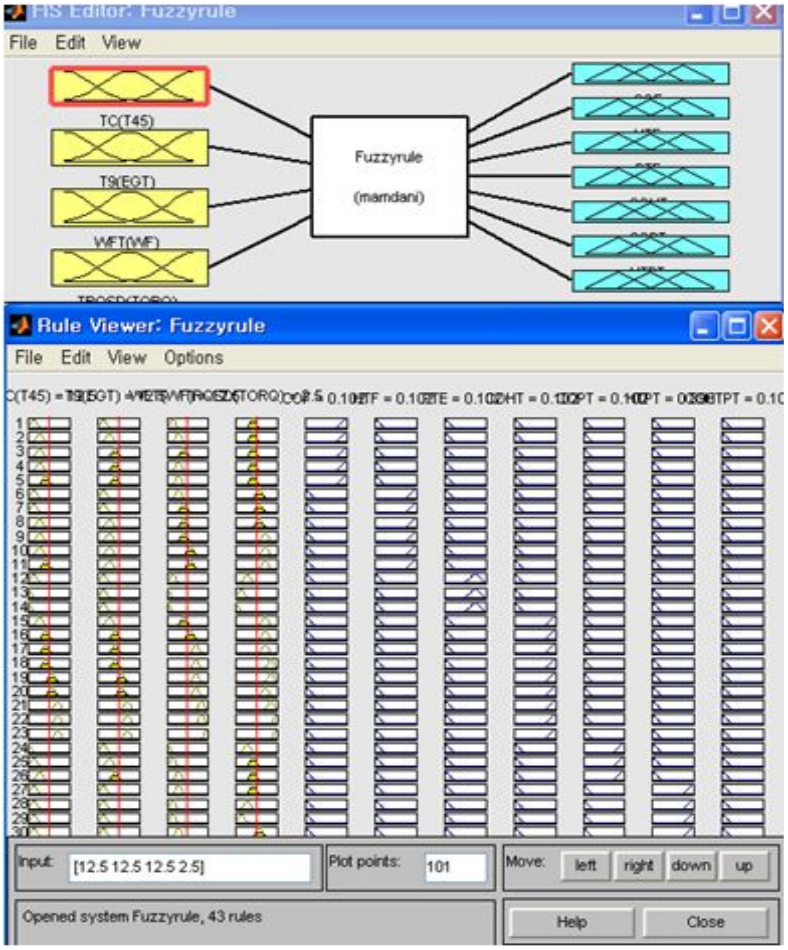


Fig. 38 MAMDANI type Fuzzy Inference System for isolating faulted components

Fuzzyfication을 위한 입력 변수는 계측 파라미터의 clean 엔진의 데이터에 대한 변화량이고 출력변수는 7가지 손상 패턴이다. 퍼지추론에는 'MAMDANI' 이론을 적용하였으며 비퍼지화에는 'Centroid' 기법을 이용하였다.

출력변수의 값은 0에서 1사이의 값을 가지며 1에 가까울수록 예측되는 손상패턴에 가깝다.

2. 구성품 손상의 정량적 진단을 위한 신경회로망 설계 및 훈련

신경회로망의 훈련은 기존에 구축한 손상 조건에 따른 성능 저하율 데이터베이스를 사용하였다. 신경회로망의 훈련에는 FFBP(Feed Forward Back Propagation) 알고리즘을 적용하였으며 Figure 39와 같이 각각 1개의 입력층과 은닉층과 출력층을 갖는다[35]. 은닉층의 전달함수에는 Tangent Sigmoid 함수, 식(15)를 적용하였으며 출력층의 전달함수에는 선형 함수, 식(16)을 적용하였다.

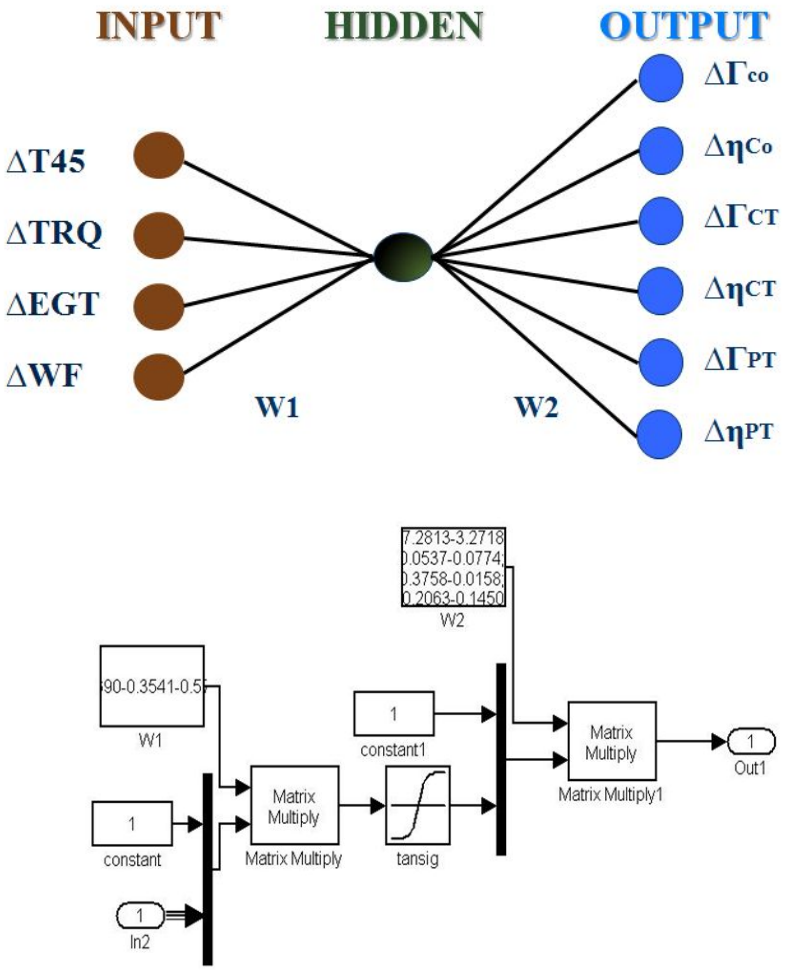


Fig. 39 Feed forward Neural Network program using SIMULINK

$$\varphi = \frac{e^{ax} - e^{ax}}{e^{ax} - e^{-ax}} \tag{15}$$

$$\varphi = x \tag{16}$$

진단의 정확성을 계산하는 RMS 오차는 식(17)과 같이 정의하였으며 신경회로망은 RMS 오차가 1% 이내에 들도록 훈련된다.

$$RMS\,error = \sqrt{\frac{\sum_{i=1}^{n}(y_n - T_n)^2}{n}} \tag{17}$$

여기에서 T는 엔진에 임의로 이식한 성능저하율이며 y는 신경회로망의 진단 결과이다.

Figure 40은 역전파 신경회로망 훈련의 알고리즘이며 Figure 41은 MATLAB/SIMULINK를 이용하여 제작한 퍼지-신경망 인공지능 진단 프로그램이다.

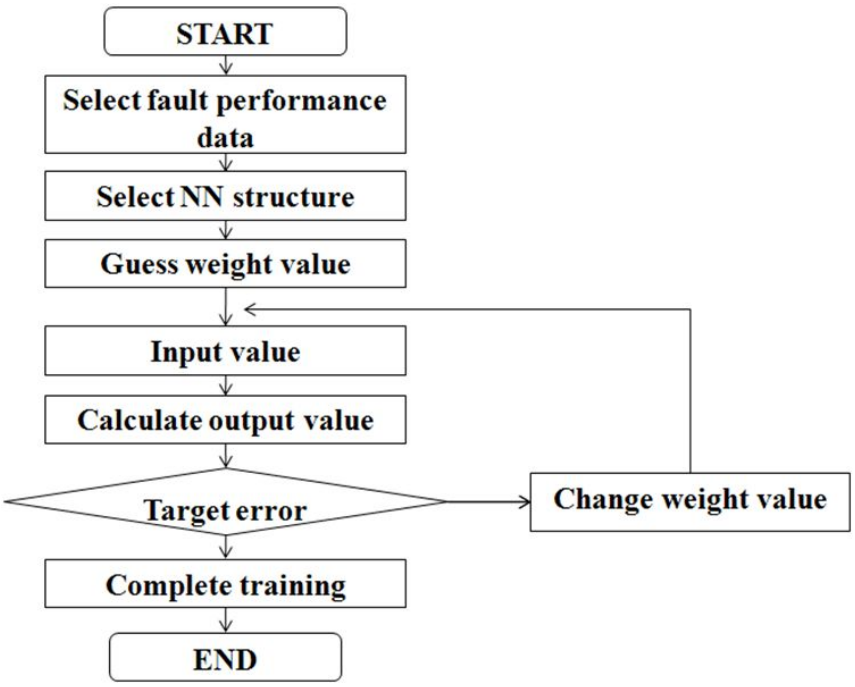


Fig. 40 Back Propagation algorithms for training Neural Network

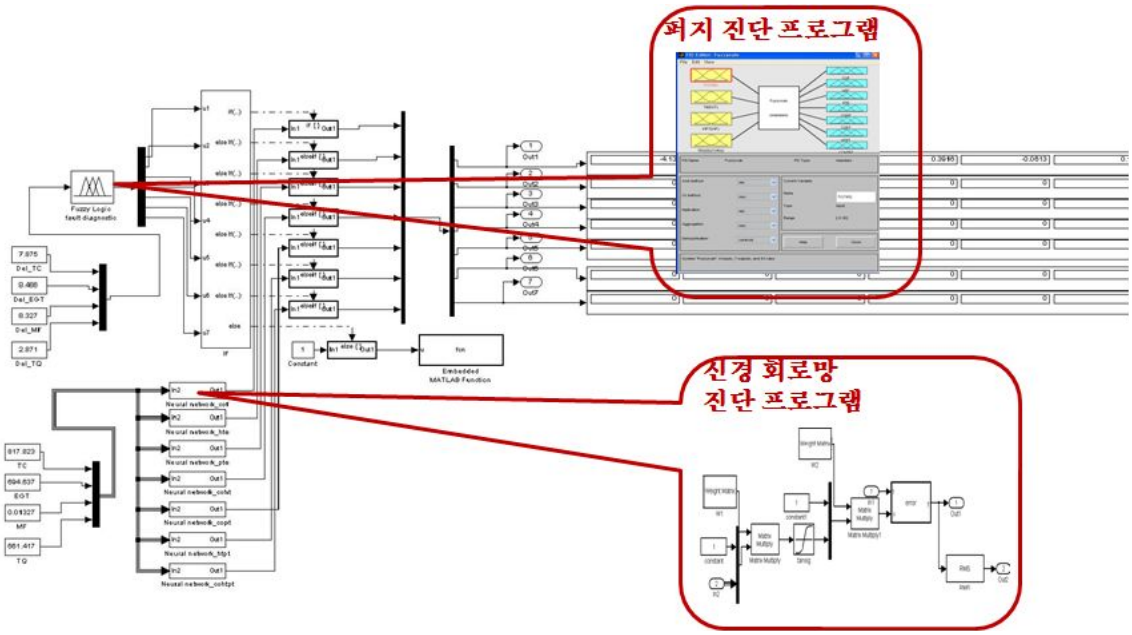


Fig. 41 Fuzzy-Neural Network Fault Diagnosis Program

3. 제작된 인공지능 진단 프로그램의 검증

제작된 인공지능 진단 프로그램을 검증하기 위해 다음과 같이 임의의 손상을 가정하였다. Table 11은 임의로 가정한 엔진 성능 저하율이다.

Table 11 Implanted fault values (IFV) of engine major components

	CO MA	CO EF	GT MA	GT EF	PT MA	PT EF
1	-5	-3	0	0	0	0
2	0	0	5	-3	0	0
3	0	0	0	0	5	-3
4	-4	-2	4	-2	0	0
5	-4	-2	0	0	4	-2
6	0	0	4	-2	4	-2
7	-5	-5	5	-5	4	-4

압축기와 가스제너레이터 터빈, 동력터빈에서 각각 오염과 침식이 발생하는 경우와 두개의 구성품에서 동시에 손상이 발생하는 경우, 모든 구성품에서 손상이 발생하는 경우에 대해 알고리즘을 적용하였다.

먼저 제작된 기준엔진 모델을 이용하여 가정된 손상량에 따라 성능해석을 수행하였다. Table 12는 각 경우의 계측변수 변화량이다.

Table 12 Measured parameter changes due to implanted faults (%)

	ITT	EGT	MF	Torque
1	7.435	8.067	8.571	2.446
2	7.817	7.027	14.367	8.231
3	-3.051	-0.933	-4.408	-6.078
4	14.385	14.072	21.714	10.588
5	5.196	7.226	5.959	-0.762
6	5.463	6.372	10.531	3.643
7	19.986	21.518	27.755	10.456

손상된 구성품을 식별하기 위해 fuzzy inference system에 적용한 결과 Table 13과 같은 결과를 얻었다.

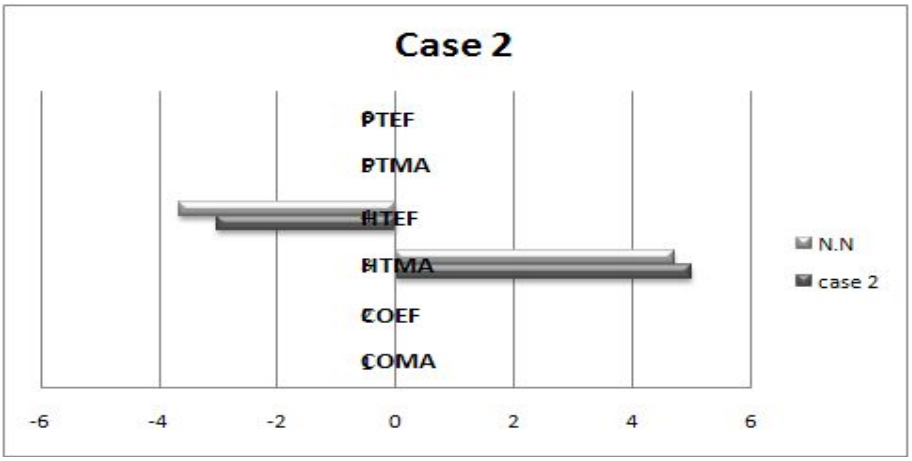
Table 13 Results of faulted components isolated by Fuzzy Inference System (IFC: Input fault cases, OFC: Output fault cases)

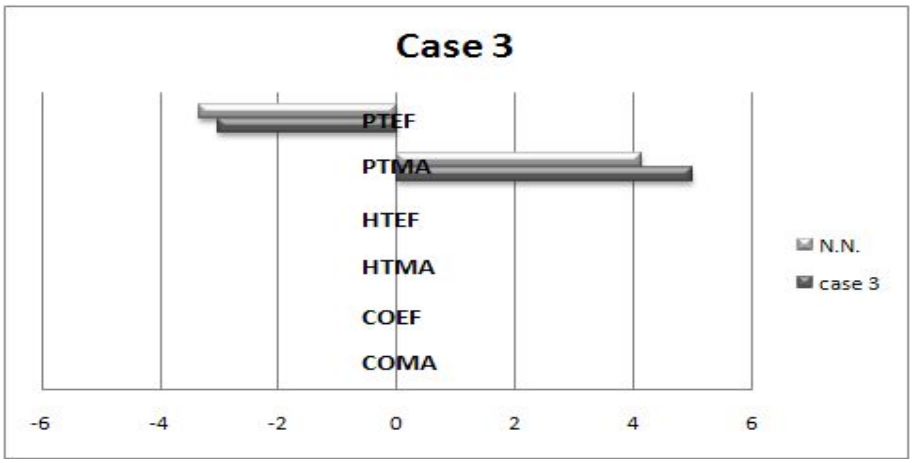
FP / case	1	2	3	4	5	6	7
1	0.51	0.091	0.089	0.087	0.431	0.263	0.092
2	0.47	0.586	0.089	0.087	0.092	0.454	0.092
3	0.098	0.091	0.688	0.087	0.092	0.085	0.092
4	0.098	0.414	0.089	0.570	0.092	0.085	0.434
5	0.401	0.091	0.207	0.087	0.568	0.085	0.092
6	0.430	0.091	0.228	0.087	0.280	0.527	0.092
7	0.453	0.270	0.089	0.453	0.280	0.473	0.566

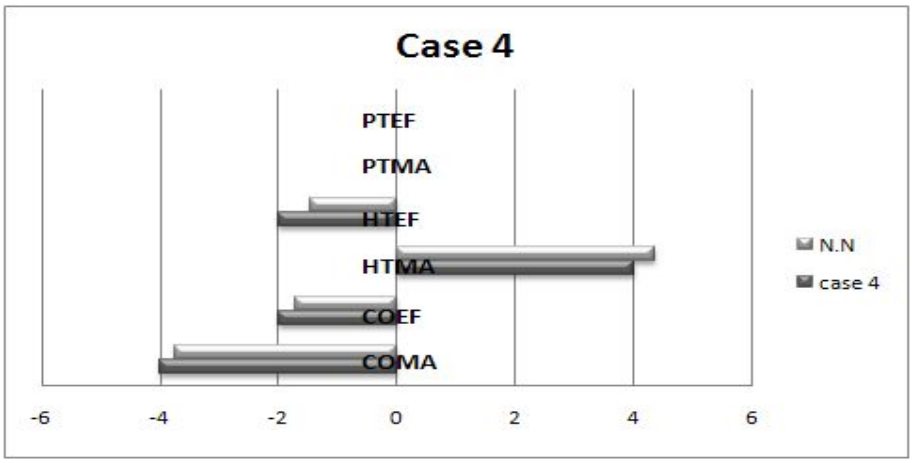
Case1의 경우 fault pattern이 1인 경우가 0.51로 가장 1에 가까운 결과를 보여 압축기 오염의 가능성이 높음을 알 수 있다. case 7의 경우에는 fault pattern이 7인 경우가 0.566으로 세 구성품에서 동시에 손상이 발생하였을 가능성이 가장 높음을 보여주고 있다.

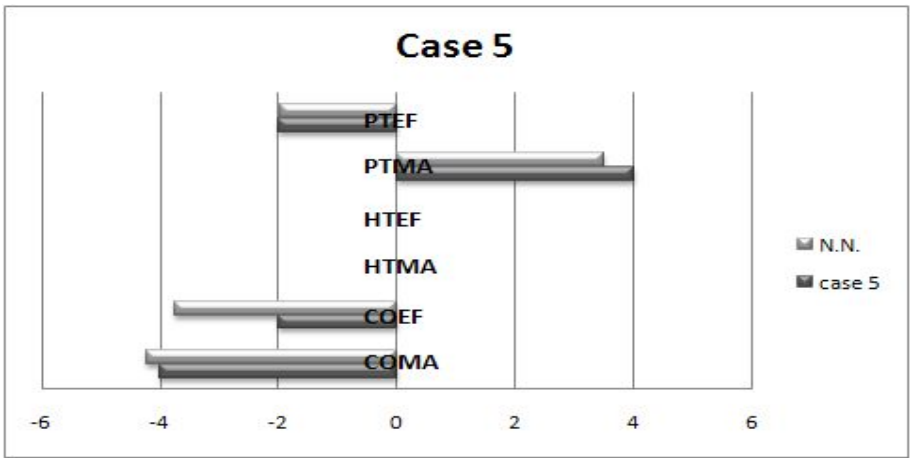
Case 2, 3, 4, 5, 6에서도 각각 fault pattern 2, 3, 4, 5, 6에서 가장 1에 가까운 값을 보여주고 있다. 이는 표 9에서 가정한 손상 패턴과 일치하는 것으로 fuzzy logc을 통해 손상된 구성품을 효과적으로 식별할 수 있음을 확인할 수 있다.

Fuzzy logic을 통해 엔진의 손상패턴이 식별되고 나면 그에 해당하는 neural network 진단 프로그램을 호출한다. 신경회로망을 이용한 손상패턴에 따른 진단 결과는 Figure 42와 같다.




Case 3
PTEF
PTMA
HTEF
HTMA
COEF
COMA
-6
-4
-2
0
2
4
6
N.N.
case 3


Case 4
PTEF
PTMA
HTEF
HTMA
COEF
COMA
-6
-4
-2
0
2
4
6
N.N
case 4


Case 5
PTEF
PTMA
HTEF
HTMA
COEF
COMA
-6
-4
-2
0
2
4
6
N.N.
case 5

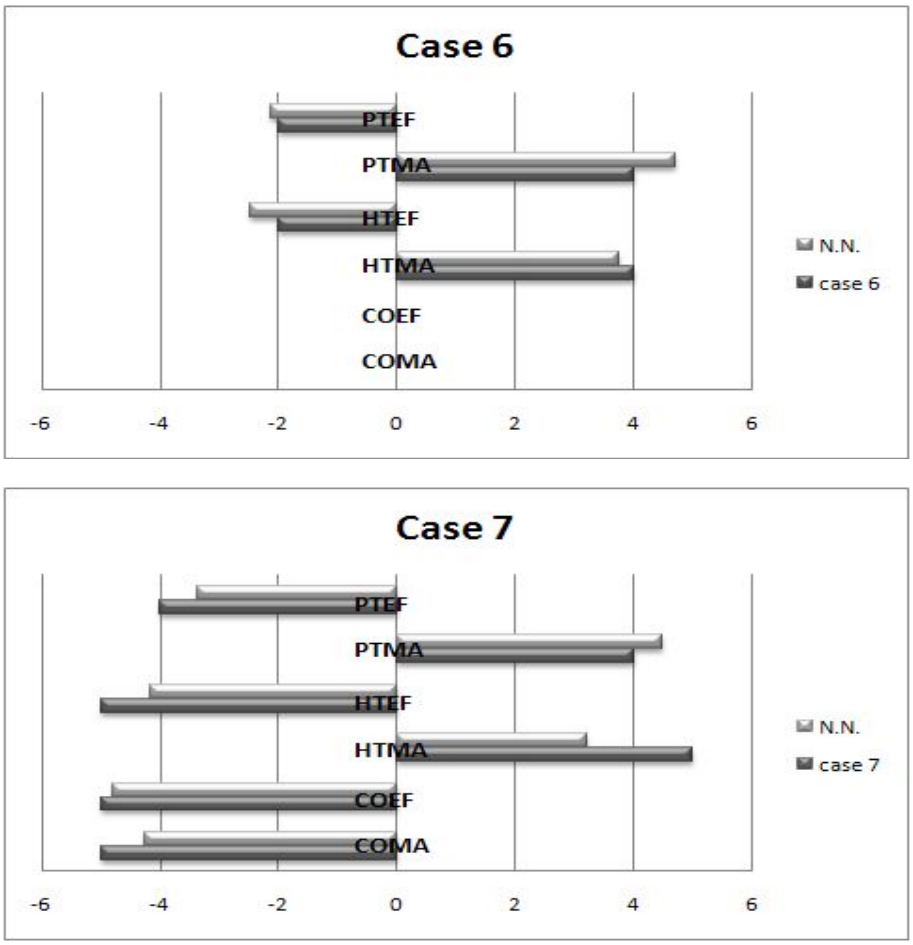


Fig. 42 Results of faulted components quantified by Neural Network diagnostic program

7가지 손상 pattern에 대한 진단을 수행한 결과 Table 11에서 임의로 준 손상값과 거의 일치하여 신뢰성 있는 진단이 가능함을 확인할 수 있었다.

제 3 절 실제 엔진 시험데이터의 진단

1. 엔진 시험 데이터 분석

엔진 시험 데이터는 설계점 기준으로 시험 수행된 결과 값으로 Figure 43과 같다.

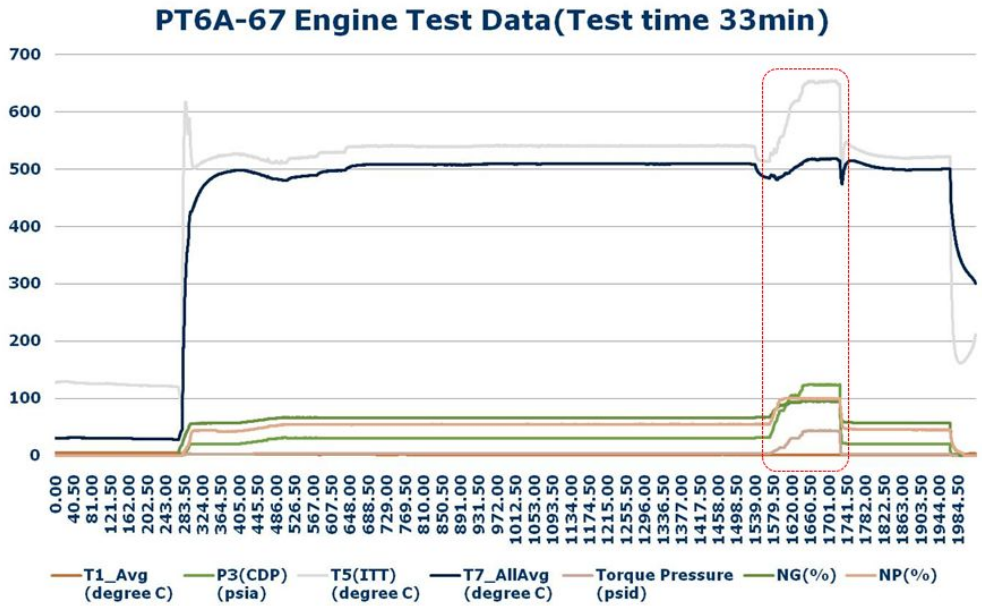


Fig. 43 PT6A-67 Test Data(33min)

초반 데이터는 시동에서 아이들 상태로 동력이 발생하는 실제 시험 데이터로 사용한 부분은 Figure 43의 빨간 사각형 안의 데이터이다.

이렇게 실제 시험 데이터는 Figure 44와 같다.

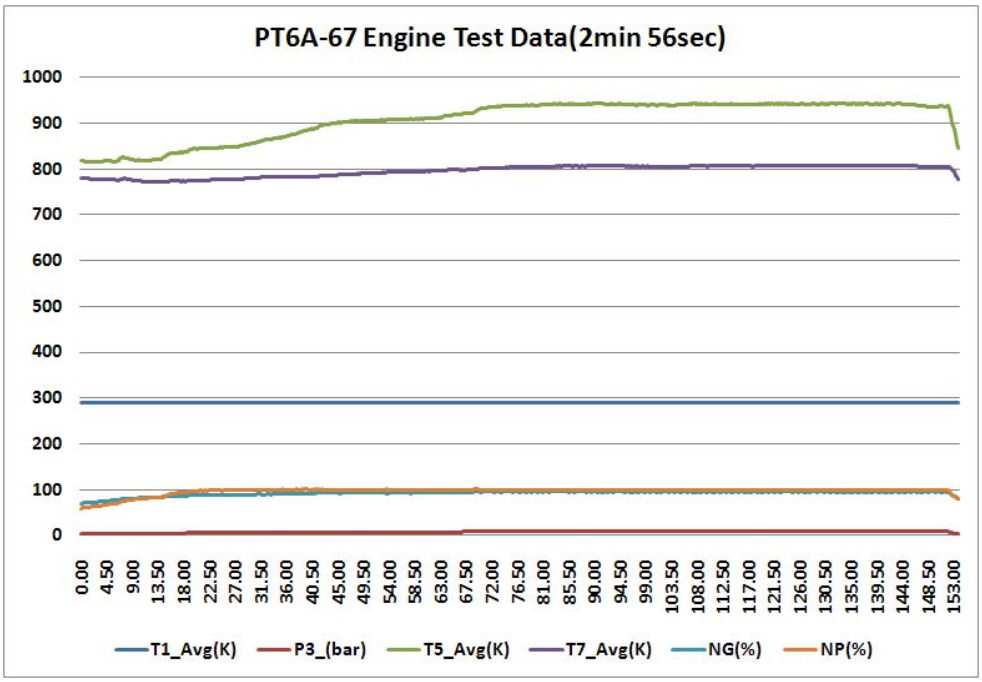


Fig. 44 PT6A-67 Test Data(3min)

이렇게 분류한 데이터는 다시 초반 부분 가스제너레이터 RPM이 70%~95%까지 증가하는 천이상태 데이터를 배제하고 95%의 정상상태 데이터를 정리하였다.

또한 엔진 시험대기 상태 입력값에 대해 같은 조건으로 미리 제작된 기준엔진 모델을 이용하여 성능 데이터 값을 계산하여 시험 데이터와 비교하였다.

Table 14는 엔진 시험 데이터와 모델을 이용하여 계산한 성능값이다.

Table 14 Test Data and Simulation Data

	P3(bar)	ITT(K)	EGT(K)	PW(KW)
Test Data	8.521	925.57	790.54	876.3
Simulation Data	8.6	965.83	805.27	841.42
%	0.92	4.2	1.83	3.98

두 데이터를 비교하면 4%이내의 경향을 보이는 것을 확인 할 수 있다. 이에 기준 모델을 이용하여 구축한 데이터베이스를 통해 제작한 인공지능 진단 프로그램을 이용하여

실제 엔진 시험 데이터를 진단 할 수 있을 것으로 판단하였다.

2. 인공지능 진단 프로그램을 이용한 엔진 시험 데이터의 진단

인공지능 진단 프로그램에는 클린엔진 식별을 위해 7가지 손상 패턴에서 모두 40% 이하 값으로 추론을 하게 되면 클린 엔진으로 판단하는 모듈을 추가하였다.

손상 판별 계수인 40%는 제작된 퍼지추론 프로그램 특성상 아무리 작은 확률에서도 20%까지 나올 수 있고 또한 예방 진단에 관해 40% 이상 확률이 최저라 판단하여 때문에 정한 수치이다.

개선된 인공지능 진단 프로그램을 이용하여 실제 엔진 시험 데이터를 진단한 결과 퍼지 추론값은 Table 15와 같다.

Table 15 Fault patterns of components

	FP1	FP2	FP3	FP4	FP5	FP6	FP7
퍼지 추론값	0.21	0.21	0.14	0.26	0.14	0.16	0.21

Table 15와 같은 퍼지 추론값을 바탕으로 인공지능 진단 프로그램은 실제 시험을 수행한 엔진이 클린 상태라고 판단하였다.

제 5 장 결 론

본 논문에서는 새로운 역식별 기법을 제안하여 무인기용 터보프롭엔진의 역모델링을 수행하여 제작된 기준엔진 모델을 이용하여 다양한 조건에서의 손상에 따른 성능저하 데이터베이스를 구축하였다. 이렇게 구축된 데이터베이스를 이용하여 퍼지룰을 생성하고 신경회로망을 훈련시켜 인공지능 진단프로그램을 제작하였다. 그리고 실제 시험 데이터를 제작된 인공지능 진단 프로그램을 이용하여 진단을 수행하였다.

본 연구의 1장에서는 추진시스템의 역모델링 및 성능모사, 상태 진단의 연구 동향을 살펴보았으며 2장에서는 선행연구로 수행한 마이크로 터보제트 엔진의 역모델링에 관해 설명하였으며 3장에서는 대상 터보프롭 엔진인 PT6A-67엔진의 역모델링에 관해 설명하였다. 그리고 마지막으로 4장에서는 퍼지로직과 신경회로망을 이용하여 진단 프로그램을 제작하여 검증하였다.

본 연구에서 제안된 역식별 기법을 사용하기 위해 먼저 마이크로 터보제트 엔진 시험장치를 구성하여 다양한 조건에서의 성능시험을 통해 성능데이터베이스를 구축하였고 이것과 유사 터보제트엔진의 성능 선도를 이용하여 마이크로 터보제트 엔진의 성능선도를 식별하였다. 또한 이 성능선도를 검증하기 위해 새로운 조건에서 엔진 성능시험을 수행하여 얻은 성능데이터와 제작된 기준 엔진 모델 프로그램 시뮬레이션 결과를 비교하였다.

그리고 실제 터보프롭 엔진을 대상 엔진으로 설정하여 다양한 조건에서 성능덱을 이용하여 성능데이터를 얻어내었고 이를 이용하여 역으로 성능 선도를 식별하였다. 이 과정에서 부족한 데이터는 유사 엔진 데이터를 참조하였다.

이렇게 역식별된 성능 선도를 이용하여 C++로 코딩하여 기준 엔진 성능 모델을 제작하였다. 제작한 성능 모델은 다시 성능덱 데이터와 비교하여 검증을 수행하였다.

이후 엔진 성능 모델을 이용하여 다양한 조건에서의 손상에 따른 성능 저하 값을 정리하여 데이터베이스를 구축하였다. 이렇게 구축된 데이터베이스를 이용하여 정성적 진단을 위한 퍼지로직의 퍼지룰을 생성하고 정량적 진단을 위한 역전파 신경회로망의 학습에 이용하였다.

완성된 인공지능 진단 프로그램은 고의로 손상값을 준 상태로 엔진 모델을 계산하여 나온 성능값을 이용하여 역으로 손상값을 찾아가는 것을 확인하는 것으로 검증하였다.

그리고 마지막으로 대상 엔진의 실제 시험 데이터를 분석하여 성능덱 데이터를 토대로 제작한 인공지능 진단 프로그램을 사용할 수 있음으로 판단하고 진단을 수행하였다.

진단 결과로 대상 엔진은 클린엔진임을 확인하였다.

Table 16 Relationship between the chapters of the thesis and the published papers

Chapters	The Published papers
Chapter 2	Kong, C.D., Lim, S.M., Kho, S.H., "Inverse Generation of Gas Turbine Component Performance Maps from Experimental Test Data", Proceeding of the 2009 APISAT Conference.
	Kong, C.D., Lim, S.M., Kim, G.W., "A STUDY ON GENERATION OF GAS TURBINE COMPONENT MAPS USING PERFORMANCE TEST DATA", Proceeding of the 2010 ICAS Conference.
	Kong, C.D., Lim, S.M., Oh, S.H., Kim, J.H., "INVERSE GENERATION OF GAS TURBINE COMPONENT PERFORMANCE MAPS FROM EXPERIMENTAL TEST DATA", Proceeding of the 2010 ASME Turbo Expo Conference.
	Kong, C.D., Lim, S.M., Kim, G.W., "Study on GUI Type Gas Turbine Performance Simulation Using Inversely Generated Component Maps and LabVIEW", Proceeding of the 2010 AJCPP Conference.
	Kong, C.D., Lim, S.M., “A Study of Inverse Modeling from Micro Gas Turbine Experimental Test Data”, Proceeding of the 2009 Journal of the Korean Society Propulsion Engineers Vol.13 No. 6, pp.1-7.
Chapter 3	Kong, C.D., Lim, S.M., Kim, G.W., Oh, S.H., Kim, J.H., “A Study of Inverse Modeling from Turboprob Engine Performance Data”, Proceeding of the 2010 KSAS Spring Conference.
	공창덕, 임세명, 김건우, 오성환, 김지현, “성능데이터를 이용한 터보프롭엔진 역모델링 연구”, 2010 국방과학연구소 40주년 기념 학술대회.
	Kong, C.D., Lim, S.M., Kim, J.H., “Study on Inverse Modeling of a Turboprop in High Altitude Operation using Engine Performance Data” Proceeding of the 2010 Journal of the Korean Society Propulsion Engineers Vol.14 No. 3, pp.16-22.

Chapters	The Published papers
Chapter 3	Kong, C.D., Lim, S.M., “Inverse Generation of Gas Turbine Component Performance Maps from Experimental Test Data”, Proceeding of the 2010 International Journal of Turbo and Jet Engines, 27, 135-148
Chapter 4	Kong, C.D., Lim, S.M., Lee, K.S., "Study on Application of Artificial Intelligent Fault Diagnostics Program to Turboprop Engine", Proceeding of the 2011 KSAS Spring Conference.
	Kong, C.D., Lim, S.M., "STUDY ON FAULT DIAGNOSTICS OF A TURBOPROP ENGINE USING INVERSE PERFORMANCE MODEL AND ARTIFICIAL INTELLIGENT METHODS", Proceeding of the 2011 ASME Turbo EXPO Conference.
	Kong, C.D., Lim, S.M., Kim, G.W., "Study on A GUI-based Condition Monitoring Program for a Turboprop Engine Using Inverse Performance Model and Nero-Fuzzy Algorithms", Proceeding of the 2011 ISABE Conference.
	Kong, C.D., Lim, S.M., Kim, G.W., "Study on Application of Artificial Intelligent Fault Diagnostics Program to Turboprop Engine", Proceeding of the 2011 IGTC Conference.
	Kong, C.D., Lim, S.M., Kim, G.W., "Study on Fault Diagnostics of a Turboprop Engine Using Fuzzy Logic and BBNN", Proceeding of the 2011 Journal of the Korean Society Propulsion Engineers Vol.15 No. 2, pp.1-7.

참 고 문 헌

1. Hormouziadis, J. and Herbig, H.. "Numerische Simulation der Kennfelder von Stromungsmaschinen." Konstrucktion, 1974, Vol. 26 : 182-186
2. Dobryanskii, G. W. and Martyanova, T. S.. "Dynamics of Aircraft Turbine Engines." 1st ed., Mashinostroenie, 1989, Moskva : 6-10
3. El-Gammal, M. A.. "An Algorithm and Criteria for Compressor Characteristics Real Time Modeling and Approximation." Journal of Engineering for Gas Turbines and Power, 1991, 113 : 112-117
4. Sieros, G., Stamatis, A. and Mathioudakis, K.. "Jet Engine Component Maps for Performance Modeling and Diagnosis." Journal of Propulsion and Power, 1997, 13 : 665- 674
5. Orkisz, M. and Stawarz, S.. "Modeling of Turbine Engine Axial-Flow Compressor and Turbine Characteristics." Engineering Transactions, 1997, 45 : 363-373
6. Kurzke, J. and Riegler, C.. "A New Compressor Map Scaling Procedure for Preliminary Conceptional Design of Gas Turbines." ASME Paper , 2000, 2000-GT-0006
7. Avva, V. S. “Effect of specimen size on the buckling behavior of laminated composites subjected to low velocity impact”, ASTM STP, 1983, 808:140-154
8. Sellers, J. F. and Daniele, C. J.. "DYNGEN - A Program for Calculating Steady-state and Transient Performance of Turbojet and Turbofan Engines." Technical Report TN-D-7901, NASA Lewis Research Center, 1975
9. MacMillan, W. L.. "Development of a Modular Type Computer Program for the Calculation of Gas Turbine Off Design Performance." Ph.D. Thesis, Cranfield University, UK, 1974
10. Fishbach, L. H. and Caddy, M. J.. "NNEP - The Navy NASA Engine

Program." Technical Report TM-X-71857, NASA Lewis Research Center, 1975

11. Palmer, J. R. and Yan, C. Z.. "TURBOTRANS - A Programming Language for the Performance Simulation of Arbitrary Gas Turbine Engines with Arbitrary Control Systems." International Journal of Turbo and Jet Engines , 1985, 19-28
12. El-Masri, M. A.. "GASCAN - An Interactive Code for Thermal Analysis of Gas Turbine Systems." Journal of Engineering for Gas Turbines and Power, 1988, Vol. 110 : 201-209
13. Bettocchi, R., Spina, P. R. and Fabbri, F.. "Dynamic Modeling of Single-Shaft Industrial Gas Turbine." ASME, 1996, 96-GT-332
14. Crosa, G., et al. "Heavy-Duty Gas Turbine Plant Aerothermodynamic Simulation using SIMULINK." Journal of Engineering for Gas Turbines and Power, 1998, Vol. 120 : 550-556
15. Grewal, M. S.. "Gas Turbine Engine Performance Deterioration Modelling and Analysis." Ph.D. Thesis, SME, Cranfield University, UK, 1988
16. Diakunchak, I. S.. "Performance Deterioration in Industrial Gas Turbines", Journal of Engineering for Gas Turbines and Power, 1992, Vol.114 : pp.161-168
17. House, P.. "Gas Path Analysis Techniques Applied to a Turboshaft Engine." MSc Thesis, SME, Cranfield University, UK, 1992
18. Escher, P. C.. "PYTHIA : An Object-Oriented Gas Path Analysis Computer Program for General Applications." Ph.D. Thesis, SME, Cranfield University, UK, 1995
19. English, L.. "Application of Gas Path Analysis, Gas Path Debris Monitoring and Expert System Technology to the Allison T56 Turboprop Engine." MSc Thesis, SME, Cranfield University, UK, 1995
20. Kim, S. K.. "Application of Neural Networks to Gas Path Analysis for Health Monitoring of Gas Turbine Engine." MSc Thesis, SME, Cranfield University, UK, 1998

21. Mathioudakis, K.. "Compressor Fault Identification from Overall Performance Data Based on Adaptive Stage Stacking." Trans. ASME Journal of Engineering for Gas Turbines and Power, 1994, 116 : 156-164
22. i-Jet 130 Manual Version 1.4, i-Complete Sdn Bhd, 2008
23. Saravanamuttoo, H.I.H., "Recommended Practices for Measurement of Gas Path Pressures and Temperatures for Performance Assessment of Aircraft Turbine Engines and Components", AGARD Advisory Report, 1990
24. Abdelwahab, M., Biesiadny, T.J. and Silver, D., "Measurement Uncertainty for the Uniform Engine Testing Program Conducted at NASA Lewis Research Center", 1987, NASA-TM-88943
25. David, D.P., "Design and Analysis of a Gas Turbine Test Facility Air System", Msc, Naval Postgraduate School, 1995
26. Walsh, P. P. and Fletcher, P., Gas Turbine Performance, 2nd ed., Blackwell Science Ltd., 2004
27. Kong, C.D., Kho, S.H., Ki, J.Y., and Jun, Y.M., "A Novel Method for Component Map Identification of a Gas Turbine Using Intelligent Method and Engine Performance Deck Data", Proceedings of ASME TURBO EXPO 2007, GT-2007-27569
28. 김용성, "Visual C++ 6 완벽가이드 2nd Edition" 영진닷컴, 2009
29. 국방과학연구소 "터보프롭엔진 역모델링 및 상태진단 연구 1차년도 보고서", 2009
30. Kong, C.D., Ki, J.Y. and Kang, M.C.. "A New Scaling Method for Component Maps Gas Turbine using System Identification", Journal of Engineering for Gas Turbines and Power Vol. 125, Copyright 2003 by ASME
31. 공창덕, 임세명. "소형 가스터빈 엔진 실험 데이터를 이용한 역모델링 연구", 한국추진공학회지, 2009, vol. 13, no. 6. pp. 1~7
32. Urban, L.A., "Gas Path Analysis Applied to Turbine Engine Condition Monitoring", Journal of Aircraft, Vol. 10, No. 7, 1973b, pp.400-406

33. Lu,P.J., Zhang,M.C., Hsu,T.C., and Zhan,J., “An Evaluation of Engine Faults Diagnostics using Artificaial Neural Networks”, Proceeding of ASME TURBO EXPO 2000, 2000-GT-0029
34. Tsoukalas, L.H. and Uhring, R.E., “Fuzzy and Neural Approaches in Engineering”, John Wiley & Sons, 1997, Inc.
35. Heykin, S., "Neural Networks - A Comprehensive Foundation", 1994, Macmilian

부 록

1. I-130 엔진 모델

```
////////////////////////////////////////////////////////////////////////////////
//////////////////// I-130 performance 2 ////////////////////////////////////////
////////////////////      BY L.S.M.      ////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////

#include<stdio.h>
#include<math.h>

void main()
{
        float COMP[5][4];
        float TUR[5][4];

        float temp;
        float TT1, T1, TT2, TT3, TT4, TT5;
        float PT1, PT2, PT3, PT4, PT5;
        float ETA1, MFP1, PR1;
        float ETA3, MFP3, PR3;
        float Ma;
        float Mf;
        float RPM;
        float TM, TTM, TZM;
        float CPA1, CPG3, CPG4;
        float gamma1, gamma3, gamma4;
        float WC, WT;
        float ETAJ=0.96;
        float ETAM=0.98;
        float del_P3=0.02;
        float R=0.287;
        float A[9], B[9];
        float per, per1, per2;
        float FAR;
        float Nz_PR;
        float Cj;
```

```
float F;
float a, b;

int i, j, k, n, m;
int imax, jmax;

A[0]=0.992313;
A[1]=0.236688;
A[2]=-1.852148;
A[3]=6.083152;
A[4]=-8.893933;
A[5]=7.097112;
A[6]=-3.234725;
A[7]=0.794571;
A[8]=-0.081873;

B[0]=-0.718874;
B[1]=8.747481;
B[2]=-15.863157;
B[3]=17.254096;
B[4]=-10.233795;
B[5]=3.081778;
B[6]=-0.36112;
B[7]=-0.003919;
B[8]=0.00;


FILE *fp_in_1;
fp_in_1=fopen("COMP.txt","r");

FILE *fp_in_2;
fp_in_2=fopen("TUR.txt","r");

FILE *fp_out;
fp_out=fopen("DATA2_OUT.txt","w");

for(i=0;i<5;i++)
{
```

```
                for(j=0;j<4;j++)
                {
                        fscanf(fp_in_1, "%f", &COMP[i][j]);
                        fscanf(fp_in_2, "%f", &TUR[i][j]);
                }
        }


        printf("TT1?\n");
        scanf("%f", &T1);

        printf("PT1?\n");
        scanf("%f", &PT1);

        printf("Mf?\n");
        scanf("%f", &Mf);

        printf("%RPM?(50~90)\n");
        scanf("%f", &RPM);

        TT1=T1+273;



// interpolation //

        for(i=0;i<5;i++)
        {
                a=COMP[i][0]*100;
                b=COMP[i+1][0]*100;
                if(RPM == a)
                {
                        ETA1=COMP[i][1];
                        MFP1=COMP[i][2];
                        PR1=COMP[i][3];
                }
                if((a<RPM)&&(b>RPM))
                {
```

```
                        ETA1=COMP[i][1]+(COMP[i+1][1]-COMP[i][1])*(RPM-a)/(b-a);
                        MFP1=COMP[i][2]+(COMP[i+1][2]-COMP[i][2])*(RPM-a)/(b-a);
                        PR1=COMP[i][3]+(COMP[i+1][3]-COMP[i][3])*(RPM-a)/(b-a);
                }
        }

        for(i=0;i<5;i++)
        {
                a=TUR[i][0]*100;
                b=TUR[i+1][0]*100;
                if(RPM == a)
                {
                        ETA3=TUR[i][1];
                        MFP3=TUR[i][2];
                        PR3=TUR[i][3];
                }
                if((a<RPM)&&(b>RPM))
                {
                        ETA3=TUR[i][1]+(TUR[i+1][1]-TUR[i][1])*(RPM-a)/(b-a);
                        MFP3=TUR[i][2]+(TUR[i+1][2]-TUR[i][2])*(RPM-a)/(b-a);
                        PR3=TUR[i][3]+(TUR[i+1][3]-TUR[i][3])*(RPM-a)/(b-a);
                }
        }

//// comp
        PT2=PT1*PR1;



        TM=TT1;
        per=0;

        while(per<0.9995)
        {
        CPA1=0;
                TZM=TM/1000;
                for(m=0;m<9;m++)
                        {
                        CPA1=CPA1+A[m]*pow(TZM, m);
```

```
                    }
            gamma1=CPA1/(CPA1-R);



            temp=pow(PR1, ((gamma1-1)/gamma1));
            TT2=TT1/ETA1*(temp-1)+TT1;
            TTM=(TT1+TT2)/2;
                    if(TM<TTM)
                            per=TM/TTM;
                    else if(TM>=TTM)
                            per=TTM/TM;
            TM=TTM;
        }
        temp=pow(TT1, 0.5);
        Ma=MFP1/temp*PT1;

        WC=CPA1*(TT2-TT1)*Ma;



///// comburster
        FAR=Mf/Ma;



///// turbine

        WT=WC/ETAM;
        PT3=PT2*(1-del_P3);

        temp=MFP3*PT3/(Ma+Mf);
        TT3=pow(temp, 2);

        PT4=PT3/PR3;



        TM=TT3;
        per1=0;
        while(per1<0.9995)
        {
```

```
        CPG3=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
                {
                CPG3=CPG3+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                }
        gamma3=CPG3/(CPG3-R);

        temp=1/PR3;
        temp=pow(temp, ((gamma3-1)/gamma3));
        temp=1-temp;
        TT4=TT3-ETA3*TT3*temp;

        TTM=(TT3+TT4)/2;
                if(TM<TTM)
                        per1=TM/TTM;
                else if(TM>=TTM)
                        per1=TTM/TM;
        TM=TTM;
}



//// nozzle

Nz_PR=PT4/PT1;

TM=TT4;
per2=0;
while(per2<0.9995)
{
        CPG4=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
                {
                CPG4=CPG4+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                }
        gamma4=CPG4/(CPG4-R);
```

```
            temp=1/Nz_PR;
            temp=pow(temp, ((gamma4-1)/gamma4));
            temp=1-temp;
            TT5=TT4-ETAJ*TT4*temp;

            TTM=(TT4+TT5)/2;
                        if(TM<TTM)
                                    per2=TM/TTM;
                        else if(TM>=TTM)
                                    per2=TTM/TM;
            TM=TTM;
        }

        temp=2*CPG4*(TT4-TT5);
        Cj=pow(temp, 0.5);

        temp=(1+Mf)*Cj;
        F=Ma*temp;


        /////
        fprintf(fp_out, "%12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f\n",
            TT2, PT2, TT3, PT3, TT4, PT4, Ma, F);


        printf("%10.5f\n", F);
}
```

2. PT6A-67엔진 모델

```
////////////////////////////////////////////////////////////////////////////////////////////////////
//////////////////////          PT6A-67 MODELING          ////////////////////////////
//////////////////////              BY L.S.M.             ////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////

#include<stdio.h>
#include<math.h>
        float CODE_1[500][500], CODE_3[500][500], CODE_4[500][500];
        float CODE_5[500][500], CODE_6[500][500], CODE_7[500][500];
        float CODE[500][500];
        int CODE_N;



        float MFP1[500], ETA1[500], PR1[500];
        float MFP4[500], ETA4[500], PR4[500];
        float MFP5[500], ETA5[500], PR5[500];
        float T0, TZ0, P0, M0;
        float TT2, TT3, TT4, TT5, TT6, TT7, T4;
        float TM, TTM, TZM;
        float TZ2, TZ3, TZ4, TZ5, TZ6, TZ7;
        float PT2, PT3, PT4, PT5, PT6, PT7, PT55=0;
        float theta, delta;
        float h, DEL_T, T00, GASG_RPM;
        float CW, GTW, PTW;
        float MaC;
        float FAR, MF, SFC;
        float LHV;
        float CPA0=0, CPA3=0, CPG4=0, CPG5=0, CPG6=0, CPAM=0, CPGB=0;
        float CPA23=0, CPG45=0, CPG56=0;
        float A[9], B[9];
        float gamma[7];
        float R;
        float temp, del_w;
        float a, b, c, d, min=1000000;
        float ANDEG, ANCAL;
```

```
        float ETAI, ETAB, DTPT, ETAT, ETAN, ETAM;
        float per, per_AN, per1, per2, per3, per4, per11, per111;
        float mv, V0, Torque=0;
        float ERR, ERRF, maxi;
        int i, j, n, m, x, y, z, q, w;
        int aa=0, ab=0, ba=0, bb=0, ca=0, cb=0;
        int imax, jmax;
        int imax1, jmax1;
        int imax2, jmax2;
        int imax3, jmax3;
        int imax4, jmax4;
        int imax5, jmax5;
        int imax6, jmax6;
        int imax7, jmax7;
        int imax8, jmax8;
        int imax9, jmax9;
//////////////////////////////////
        int k, kmx;

void main()
{
        FILE *fp_in_1;
        fp_in_1=fopen("CODE_1.txt","r");
        FILE *fp_in_2;
        fp_in_2=fopen("CODE_3.txt","r");
        FILE *fp_in_3;
        fp_in_3=fopen("CODE_4.txt","r");

        FILE *fp_in_4;
        fp_in_4=fopen("CODE_5.txt","r");
        FILE *fp_in_5;
        fp_in_5=fopen("CODE_6.txt","r");
        FILE *fp_in_6;
        fp_in_6=fopen("CODE_7.txt","r");

        FILE *fp_out_7;
        fp_out_7=fopen("performance.txt","w");
```

```
////////////////////////////

        FILE *fp_in_7;
        fp_in_7=fopen("abab.txt","r");
        FILE *fp_out_1;
        fp_out_1=fopen("itt.txt","w");
        FILE *fp_out_2;
        fp_out_2=fopen("egt.txt","w");
        FILE *fp_out_3;
        fp_out_3=fopen("mf.txt","w");
        FILE *fp_out_4;
        fp_out_4=fopen("torque.txt","w");

///////////  Fixed Input Data////////
        A[0]=0.992313;
        A[1]=0.236688;
        A[2]=-1.852148;
        A[3]=6.083152;
        A[4]=-8.893933;
        A[5]=7.097112;
        A[6]=-3.234725;
        A[7]=0.794571;
        A[8]=-0.081873;

        B[0]=-0.718874;
        B[1]=8.747481;
        B[2]=-15.863157;
        B[3]=17.254096;
        B[4]=-10.233795;
        B[5]=3.081778;
        B[6]=-0.36112;
        B[7]=-0.003919;
        B[8]=0.00;

        ETAI=0.9;
        ETAB=0.925;
        DTPT=0.03;
        ETAN=0.96;
```

```
ETAM=0.96;
ANDEG=0.06064504;
R=0.28705;
LHV=42769.84;

//////////  File In  ////////////
fscanf(fp_in_1, "%d", &imax1);
fscanf(fp_in_1, "%d", &jmax1);

fscanf(fp_in_2, "%d", &imax2);
fscanf(fp_in_2, "%d", &jmax2);

fscanf(fp_in_3, "%d", &imax3);
fscanf(fp_in_3, "%d", &jmax3);


fscanf(fp_in_4, "%d", &imax4);
fscanf(fp_in_4, "%d", &jmax4);

fscanf(fp_in_5, "%d", &imax5);
fscanf(fp_in_5, "%d", &jmax5);

fscanf(fp_in_6, "%d", &imax6);
fscanf(fp_in_6, "%d", &jmax6);



for(i=0;i<imax1;i++)
{
        for(j=0;j<jmax1;j++)
        {
                fscanf(fp_in_1, "%f", &CODE_1[i][j]);
        }
}

for(i=0;i<imax2;i++)
{
        for(j=0;j<jmax2;j++)
```

```
		{
			fscanf(fp_in_2, "%f", &CODE_3[i][j]);
		}
}
for(i=0;i<imax3;i++)
{
		for(j=0;j<jmax3;j++)
		{
			fscanf(fp_in_3, "%f", &CODE_4[i][j]);
		}
}




for(i=0;i<imax4;i++)
{
		for(j=0;j<jmax4;j++)
		{
			fscanf(fp_in_4, "%f", &CODE_5[i][j]);
		}
}

for(i=0;i<imax5;i++)
{
		for(j=0;j<jmax5;j++)
		{
			fscanf(fp_in_5, "%f", &CODE_6[i][j]);
		}
}

for(i=0;i<imax6;i++)
{
		for(j=0;j<jmax6;j++)
		{
			fscanf(fp_in_6, "%f", &CODE_7[i][j]);
		}
}
```

```
////////////////////////////////////////
        fscanf(fp_in_7, "%d", &kmx);
        for(k=1;k<=kmx;k++)
                {
                 ////////////////////
                 // FLIGHT CONDITION //
                 ////////////////////
        /*      while(1)
                {
                printf("FLIGHT ALTITUDE(0~29999m)\n");
                scanf("%f", &h);
                if((0<=h)&&(h<30000))
                        break;
                else
                        printf("WRONG INPUT DATA\n");
                }*/

                h=0;  //alt

        //      printf("MFPC(-5~5%)\n");
        //      scanf("%d", &aa);

        //  aa=0;

        //      printf("ETAC(-5~5%)\n");
        //      scanf("%d", &ab);

        //  ab=0;

        //      printf("MFPCT(-5~5%)\n");
        //      scanf("%d", &ba);

        //      ba=0;

        //      printf("ETACT(-5~5%)\n");
        //      scanf("%d", &bb);

        //      bb=0;
```

```
//////////////////////////////////////////////

            fscanf(fp_in_7, "%d  %d", &aa, &ab);
            fscanf(fp_in_7, "%d  %d", &ba, &bb);
            fscanf(fp_in_7, "%d  %d", &ca, &cb);

//          printf("MFPPT(-5~5%)\n");
//          scanf("%d", &ca);

//  ca=0;

//          printf("ETAPT(-5~5%)\n");
//          scanf("%d", &cb);

//  cb=0;

//          printf("DELTA T?\n");
//          scanf("%f", &DEL_T);

  DEL_T=0;

/*          while(1)
            {
            printf("FLIGHT MACH NUMBER\n");
            scanf("%f", &M0);
            if(0<=M0)
                        break;
            else
                        printf("WRONG INPUT DATA\n");
            }
*/
  M0=0;

/*          while(1)
            {
            printf("FLIGHT CONDITION CODE NUMBER(1,3,4,5,6,7)\n");
            scanf("%d", &CODE_N);
            if((CODE_N==1)||(CODE_N==3)||(CODE_N==4)||(CODE_N==5)||(CODE_N==6)||(CODE_N==7))
```

```
                break;
        else
                printf("WRONG INPUT DATA\n");
        }

*/  CODE_N=1;

        if(CODE_N==1)
        {
                for(i=0;i<imax1;i++)
                        {
                                for(j=0;j<jmax1;j++)
                                {
                                CODE[i][j]=CODE_1[i][j];
                                imax=imax1;
                                jmax=jmax1;
                                }
                        }
        }
        else if(CODE_N==3)
        {
                for(i=0;i<imax2;i++)
                        {
                                for(j=0;j<jmax2;j++)
                                {
                                CODE[i][j]=CODE_3[i][j];
                                imax=imax2;
                                jmax=jmax2;
                                }
                        }
        }
        else if(CODE_N==4)
        {
                for(i=0;i<imax3;i++)
                        {
                                for(j=0;j<jmax3;j++)
                                {
                                CODE[i][j]=CODE_4[i][j];
```

```
                    imax=imax3;
                    jmax=jmax3;
                    }
            }
}
else if(CODE_N==5)
{
        for(i=0;i<imax4;i++)
            {
                    for(j=0;j<jmax4;j++)
                    {
                    CODE[i][j]=CODE_5[i][j];
                    imax=imax4;
                    jmax=jmax4;
                    }
            }
}
else if(CODE_N==6)
{
        for(i=0;i<imax5;i++)
            {
                    for(j=0;j<jmax5;j++)
                    {
                    CODE[i][j]=CODE_6[i][j];
                    imax=imax5;
                    jmax=jmax5;
                    }
            }
}
else if(CODE_N==7)
{
        for(i=0;i<imax6;i++)
            {
                    for(j=0;j<jmax6;j++)
                    {
                    CODE[i][j]=CODE_7[i][j];
                    imax=imax6;
                    jmax=jmax6;
```

```
                                        }
                                }
}

if(h<=11000)
{
T0=288.15-0.0065*h+DEL_T;
T00=288.15-0.0065*h;
P0=pow(T00/288.15, 5.2561)*1.01325;
}

else if((11000<h)||(h<24994))
{
T0=216.67+DEL_T;
temp=0.000157689*(h-10998.1);
temp=exp(temp);
P0=0.2263253/temp;
}

else if((24994<h)||(h<30000))
{
T0=216.65+0.0029892*(h-24994)+DEL_T;
T00=216.65+0.0029892*(h-24994);
P0=2.5237*pow(216.65/T00, 11.8);
}

TZ0=T0/1000;

for(m=0;m<9;m++)
{
        CPA0=CPA0+A[m]*pow(TZ0, m);
}

gamma[0]=CPA0/(CPA0-R);


mv=pow((gamma[0]*R*1000*T0), 0.5);
```

```
V0=mv*M0;

 ////////////////////////////
 ////////// INLET ////////////
 ////////////////////////////

TT2=T0+(V0*V0)/(2*CPA0*1000);

temp=1+(ETAI*(V0*V0)/(2*CPA0*1000*T0));
PT2=pow(temp, (gamma[0]/(gamma[0]-1))) * P0;

temp=(1+((gamma[0]-1)/2)*pow(M0, 2));
theta=(T0/288.15)*temp;
delta=(P0/1.01325)*pow(temp, (gamma[0]/(gamma[0]-1)));


 ////////////////////////////////////
 ////////// COMPRESSURE ////////////
 ////////////////////////////////////

for(i=0;i<imax;i++)
{

        TM=TT2;
        per=0;
        while(per<0.9999995)
        {
                CPA3=0;
                TZM=TM/1000;
                for(m=0;m<9;m++)
                        {
                        CPA3=CPA3+A[m]*pow(TZM, m);
                        }
                gamma[3]=CPA3/(CPA3-R);


                temp=pow(CODE[i][1],((gamma[3]-1)/gamma[3]));
                TT3=(TT2*(temp-1)/(CODE[i][2]/*+CODE[i][2]*ab/100*/))+TT2;
```

```
                TTM=(TT2+TT3)/2;
                        if(TM<TTM)
                                per=TM/TTM;
                        else if(TM>=TTM)
                                per=TTM/TM;
                TM=TTM;
        }
        PT3=CODE[i][1]*PT2;

        MaC=(CODE[i][0]/*+(CODE[i][0]*aa/100)*/)*(P0*100)/pow(T0, 0.5);

        CW=MaC*CPA3*(TT3-TT2)/0.96;


    ////////////////////////////////
    ////////// COMBUSTOR ////////////
    ////////////////////////////////

        PT4=PT3*(1-DTPT);


    ////////////////////////////////////////////
    ////////// GAS GENERATER TURBINE //////////
    ////////////////////////////////////////////

//      per111=0;
//      while(per111<0.99995)
//      {
                w=1;
                MF=0;
                per11=0;
                T4=0;
                FAR=0;
                while(per11<0.9999995)
                {
                        temp=(CODE[i][3]/*+CODE[i][3]*ba/100*/)*PT4*100/(MaC+MF);
                        TT4=pow(temp, 2);
```

```
TM=(TT3+TT4)/2;

CPGB=0;
TZM=TM/1000;
for(m=0;m<9;m++)
        {
        C P G B = C P G B + A [ m ] * p o w ( T Z M ,
m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
        }

FAR=CPGB*(TT4-TT3)/ETAB/LHV;
MF=FAR*MaC;

if(TT4<T4)
        per11=TT4/T4;
else if(TT4>=T4)
        per11=T4/TT4;

T4=TT4;
}


TM=TT4;
per1=0;
while(per1<0.9999995)
{
        CPG4=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
        {
        C P G 4 = C P G 4 + A [ m ] * p o w ( T Z M ,
m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
        }
        gamma[4]=CPG4/(CPG4-R);


        temp=pow((1/CODE[i][4]),((gamma[4]-1)/gamma[4]));
```

```
                    TT5=TT4-(CODE[i][5]/*+CODE[i][5]*bb/100*/)*TT4*(1-temp);
                    TTM=(TT4+TT5)/2;
                        if(TM<TTM)
                            per1=TM/TTM;
                        else if(TM>=TTM)
                            per1=TTM/TM;
                    TM=TTM;
                }

//              PT5=PT4/CODE[i][4];
            PT5=(MaC+MF)*pow(TT5, 0.5)/(CODE[i][6]/*+CODE[i][6]*ca/100*/)/100;

/*              if(PT5<PT55)
                    per111=PT5/PT55;
                else if(PT5>=PT55)
                    per111=PT55/PT5;

                PT4=PT55*CODE[i][4];

            }*/
                GTW=CPG4*(TT4-TT5)*(MaC+MF);


        ////////////////////////////////////
        ////////// POWER TURBINE ////////////
        ////////////////////////////////////

            TM=TT5;
            per2=0;
            while(per2<0.9999995)
            {
                CPG5=0;
                TZM=TM/1000;
                for(m=0;m<9;m++)
                    {
                    CPG5=CPG5+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                    }
                gamma[5]=CPG5/(CPG5-R);
```

```
                temp=pow((1/CODE[i][7]),((gamma[5]-1)/gamma[5]));
                TT6=TT5-(CODE[i][8]/*+CODE[i][8]*cb/100*/)*TT5*(1-temp);
                TTM=(TT5+TT6)/2;
                        if(TM<TTM)
                                per2=TM/TTM;
                        else if(TM>=TTM)
                                per2=TTM/TM;
                TM=TTM;
        }

        PT6=PT5/CODE[i][7];

        PTW=CPG5*(TT5-TT6)*(MaC+MF);

        SFC=3600*MF/PTW;

        if(CW>=GTW)
        {
                del_w=CW-GTW;
        }
        else if(CW<GTW)
        {
                del_w=GTW-CW;
        }


        if(min>del_w)
        {
                min=del_w;
                maxi=i;
        }

        }
```

//

////////////////////////////////////

```
////////// COMPRESSURE ////////////
////////////////////////////////////
                i=maxi;
//          for(i=0;i<imax;i++)
            {

                TM=TT2;
                per=0;
                while(per<0.9999995)
                {
                        CPA3=0;
                        TZM=TM/1000;
                        for(m=0;m<9;m++)
                                {
                                CPA3=CPA3+A[m]*pow(TZM, m);
                                }
                        gamma[3]=CPA3/(CPA3-R);


                        temp=pow(CODE[i][1],((gamma[3]-1)/gamma[3]));
                        TT3=(TT2*(temp-1)/(CODE[i][2]+CODE[i][2]*ab/100))+TT2;
                        TTM=(TT2+TT3)/2;
                                if(TM<TTM)
                                        per=TM/TTM;
                                else if(TM>=TTM)
                                        per=TTM/TM;
                        TM=TTM;
                }
                PT3=CODE[i][1]*PT2;

                MaC=(CODE[i][0]+CODE[i][0]*aa/100)*(P0*100)/pow(T0, 0.5);

                CW=MaC*CPA3*(TT3-TT2)/0.96;


////////////////////////////////
////////// COMBUSTOR ////////////
////////////////////////////////
```

```
PT4=PT3*(1-DTPT);


////////////////////////////////////////////
////////// GAS GENERATER TURBINE //////////
////////////////////////////////////////////

w=1;
MF=0;
per11=0;
T4=0;
FAR=0;
while(per11<0.9999995)
{
        temp=(CODE[i][3]+CODE[i][3]*ba/100)*PT4*100/(MaC+MF);
        TT4=pow(temp, 2);



        TM=(TT3+TT4)/2;

        CPGB=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
                {
                CPGB=CPGB+A[m]*pow(TZM,        m)+(FAR/(1+FAR)*(B[m]*pow(TZM,
m)));

                }

        FAR=CPGB*(TT4-TT3)/ETAB/LHV;
        MF=FAR*MaC;

        if(TT4<T4)
                per11=TT4/T4;
        else if(TT4>=T4)
                per11=T4/TT4;

        T4=TT4;
```

```
                }


                TM=TT4;
                per1=0;
                while(per1<0.9999995)
                {
                        CPG4=0;
                        TZM=TM/1000;
                        for(m=0;m<9;m++)
                                {
                                CPG4=CPG4+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                                }
                        gamma[4]=CPG4/(CPG4-R);


                        temp=pow((1/CODE[i][4]),((gamma[4]-1)/gamma[4]));
                        TT5=TT4-(CODE[i][5]+CODE[i][5]*bb/100)*TT4*(1-temp);
                        TTM=(TT4+TT5)/2;
                                if(TM<TTM)
                                        per1=TM/TTM;
                                else if(TM>=TTM)
                                        per1=TTM/TM;
                        TM=TTM;
                }

//              PT5=PT4/CODE[i][4];
                PT5=(MaC+MF)*pow(TT5, 0.5)/(CODE[i][6]+CODE[i][6]*ca/100)/100;

                        GTW=CPG4*(TT4-TT5)*(MaC+MF);


        ////////////////////////////////////
        ////////// POWER TURBINE ////////////
        ////////////////////////////////////

                TM=TT5;
                per2=0;
```

```
while(per2<0.9999995)
{
        CPG5=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
                {
                CPG5=CPG5+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                }
        gamma[5]=CPG5/(CPG5-R);



        temp=pow((1/CODE[i][7]),((gamma[5]-1)/gamma[5]));
        TT6=TT5-(CODE[i][8]+CODE[i][8]*cb/100)*TT5*(1-temp);
        TTM=(TT5+TT6)/2;
                if(TM<TTM)
                        per2=TM/TTM;
                else if(TM>=TTM)
                        per2=TTM/TM;
        TM=TTM;
}

PT6=PT5/CODE[i][7];

PTW=CPG5*(TT5-TT6)*(MaC+MF);

//////////////////////////////
////////// Nozzle ////////////
//////////////////////////////


TM=TT6;
per3=0;
while(per3<0.9999995)
{
        CPG6=0;
        TZM=TM/1000;
        for(m=0;m<9;m++)
                {
```

```
                    CPG6=CPG6+A[m]*pow(TZM, m)+(FAR/(1+FAR)*(B[m]*pow(TZM, m)));
                    }
            gamma[6]=CPG6/(CPG6-R);



            temp=pow((P0/PT6),((gamma[6]-1)/gamma[6]));
            TT7=TT6-0.96*TT6*(1-temp);
            TTM=(TT6+TT7)/2;
                    if(TM<TTM)
                            per3=TM/TTM;
                    else if(TM>=TTM)
                            per3=TTM/TM;
            TM=TTM;
        }

////////////////////////////////
////////// Perfomance /////////////
////////////////////////////////

        SFC=3600*MF/PTW;

        Torque=9549.3*PTW/1700;

/*      if(CW>=GTW)
        {
                del_w=CW-GTW;
        }
        else if(CW<GTW)
        {
                del_w=GTW-CW;
        }
        ETAM=CW/GTW;
        if(del_w==min)
        {
                break;
        }
*/
```

```
                    }



                    ////////////////////////////
                    ////////// PRINT ////////////
                    ////////////////////////////

                    fprintf(fp_out_7, "%12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f    %12.5f
    %12.5f    %12.5f    %12.5f    %12.5f\n",
                              TT3, PT3, TT4, PT4, TT5, PT5, TT6, PT6, MaC, MF, CW, GTW, PTW, SFC);

                    printf("%12.5f \n", TT5);
                    printf("%12.5f \n", TT7);
                    printf("%12.5f \n", MF);
                    printf("%12.5f \n", Torque);

                    fprintf(fp_out_1,"%12.5f \n\n\n", TT5);
                    fprintf(fp_out_2,"%12.5f \n\n\n", TT7);
                    fprintf(fp_out_3,"%12.5f \n\n\n", MF);
                    fprintf(fp_out_4,"%12.5f \n\n\n", Torque);
          //        printf("%12.5f \n", (CODE[i][0]+(CODE[i][0]*aa/100)));
          //        printf("%12.5f \n", CODE[i][0]);
          //        printf("%12.5f \n", CPG6);
          //        printf("%12.5f \n", FAR);
          }
}
```

2. 역전파 신경망 훈련 프로그램

```
% ========================================================================
% NEURAL NETWORK DESIGN FOR GAS TURBINE FAULT DETECTION
% PROGRAMMED BY Lim Se Myeong
% 2010. 10. 11.
% ========================================================================

% HIDDEN LAYER : TANSIGMOID
% OUTPUT LAYER : LINEAR

% INITIAL VALUE
function[]=fileload
prompt={'File number?'}
filenum=inputdlg(prompt, 'input window');
filenum=str2double(filenum)
if(filenum==1)
    load 'co.txt';
    data=co;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==2)
    load 'ct.txt';
    data=ct;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==3)
    load 'pt.txt';
    data=pt;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==4)
    load 'coct.txt';
    data=coct;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==5)
    load 'copt.txt';
```

```
    data=copt;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==6)
    load 'ctpt.txt';
    data=ctpt;
    t=data(:,5:10);
    x=data(:,1:4);
elseif(filenum==7)
    load 'coctpt.txt';
    data=coctpt;
    t=data(:,5:10);
    x=data(:,1:4);
end
t=t';
x=x';



maxcycles=1000;
RMS_goal=1.5;
lr=0.1;
hidden=1;
a=0.5;                    % 기울기 parameter

% DATA BASE LOAD

%[x,t] = DB_fault_cof;
%[x,t] = DB_fault_hte;
%[x,t] = DB_fault_pte;
%[x,t] = DB_fault_coht;
%[x,t] = DB_fault_copt;
%[x,t] = DB_fault_htpt;
%[x,t] = DB_fault_cohtpt;
```

```
[inputs, patterns1]=size(x);
[outputs, patterns]=size(t);

if patterns1 ~= patterns
    error('입력 패턴과 목표 패턴이 같아야 함')
end

% WEIGHT VALUE INITIALIZE

W1=0.1*randn(hidden, inputs+1) % 출력층 가중 행렬의 초기화
W2=0.1*randn(outputs, hidden+1)% 은닉층 가중 행렬의 초기화

% LEARNING

[hidden, inputs1]=size(W1);
[outputs1, hidden1]=size(W2);

if outputs1 ~= outputs
    error('W1에 관한 차원이 목표 벡터의 것과 일치하지 않음')
end
if inputs1 ~= inputs+1
    error('W2에 관한 차원이 입력벡터와 일치하지 않음')
end
if hidden1~=hidden+1
    error('W1과 W2의 차원이 일치하지 않음')
end

terms=outputs*patterns;
RMS=zeros(1,maxcycles);
LR=zeros(1,maxcycles);
X=[ones(1,patterns);x];
h=tansig(W1*X);
x=size(h);
H=[ones(1,patterns);h];
output=linearx(W2*H);
e=t-output;
RMS(1)=sqrt(sum(sum(e.^2))/terms);
```

```
fprintf('Epoch=1, lr=%f,RMS Error=%f\n',lr,RMS(1));

for i=2:maxcycles
    delta2=e*10;
    del_W2=lr*delta2*H';
    delta1=a*(h.*(1-h)).*(W2(:,2:hidden+1)'*delta2);
    del_W1=lr*delta1*X';
    new_W1=W1+del_W1;
    new_W2=W2+del_W2;
    new_h=tansig(new_W1*X);
    new_H=[ones(1,patterns);new_h];
    new_output=linearx(new_W2*new_H);
    new_e=t-new_output;
    RMS(i)=sqrt(sum(sum(new_e.^2))/terms);
    if RMS(i)<RMS(i-1)
        W1=new_W1;
        W2=new_W2;
        e=new_e*10;
        h=new_h;
        H=new_H;
        lr=lr*1.05;
    else lr=lr*0.5;
    end

    LR(i)=lr;

    if RMS(i)<RMS_goal;
     break; end

    if rem(i,1)==0
        fprintf('Epoch=%i,lr=%f,RMS Error=%f\n',i,LR(i),RMS(i));
    end
end

RMS=nonzeros(RMS);
LR=nonzeros(LR);

%save result_cof W1 W2 RMS LR %checkfunction[]=fileload
```

```
if(filenum==1)
    save result_co W1 W2 RMS LR
elseif(filenum==2)
    save result_ct W1 W2 RMS LR
elseif(filenum==3)
    save result_pt W1 W2 RMS LR
elseif(filenum==4)
    save result_coct W1 W2 RMS LR
elseif(filenum==5)
    save result_copt W1 W2 RMS LR
elseif(filenum==6)
    save result_ctpt W1 W2 RMS LR
elseif(filenum==7)
    save result_coctpt W1 W2 RMS LR
end
t=t';
x=x';



% RESULT PLOT
plot(i, RMS);
semilogy(RMS);
title('Back Propergation Training Result');
xlabel('Number of Iteration');
ylabel('RMS Error');

% MAX. CYCLE CHECK
cycles=length(RMS);
if cycles == maxcycles
    fprintf('\n\n *** BP Training Terminate!')
    fprintf('\n *** RMS = %e     \n\n',min(RMS))
else
    fprintf('\n *** %0.f epoch, Good Training!!! ***',cycles)
    fprintf('\n *** RMS = %e     \n\n',min(RMS))
end
```

감사의 글

정말 많은 분들 덕분에 무사히 논문을 완성할 수 있었습니다. 가장 먼저 학업을 계속할 수 있도록 후원을 아끼지 않고 응원해 주신 부모님께 감사드립니다. 그리고 미숙한 제게 학문에 가르침을 주시며 지도를 해주신 공창덕 교수님께 깊은 감사를 드립니다. 또한 바쁘신 와중에도 시간을 내어 교육 해주신 기자영 선배님과 고성희 선배님, 강명철 선배님, 노홍석 선배님께 감사드립니다. 연구 방향을 결정하는 데 방향을 제시해주시고 또한 여러 가지 도움을 주신 국방과학 연구소의 오성환 박사님과 김지현 선임연구원님에게 감사드립니다.

함께 실험실 생활을 한 이들을 한명씩 생각해보면 가장 먼저 실험실 책임자로서 후배들을 잘 챙겨주고 많은 것을 알려준 현범 형, 고마워요. 그리고 역시 함께 공부하며 1년 먼저 졸업한 선배이자 친구 영주, 목표로 하는 것 꼭 이루길 바래. 무엇보다도 프로그램 언어 쪽으로는 아무것도 모르는 제게 기초부터 많은 것을 가르쳐준 명국, 고마워.

경선, 민웅, 성진 남은 기간 동안 논문 준비 잘해서 꼭 성공해라.

연소 실험실의 김태규 교수님, 이런 저런 새로운 지식과 시야를 넓힐 수 있는 가르침을 주셔서 감사합니다.

그리고 대일, 매번 스캔과 복사 부탁 들어줘서 고맙다. 남은 기간 잘 준비해서 학위논문 잘 써라.

유체 실험실의 지수와 재경이도 역시 남은 기간 마무리 잘해.

길수와 동빈이도 처음 생각 했던 데로 진학 잘 해서 의미 있는 배움의 시간을 보냈으면 한다.

마지막으로 함께 졸업하는 하승이와 건우, 수고했다.