금융통계 및 시계열 분석 제2강 요약

스크립트 프로그래밍

함수

- `f.name <- function(param1=default1, param2=default2, ...) { expression }`
- `...` : 가변길이 변수. `list(...)`로 해석
- 마지막 expression을 return

흐름 제어

- `if (condition) { true-expression } else { false-expression}` : 조건 분기
- `ifelse(vector, true-value, false-value)` : 벡터 조건 처리
- `switch (expression, input1=output1, input2=output2)` : 복수 조건 분기
- `for (var in sequence) { expression }` : 반복
- `while (condition) { expression }` : 조건 반복
- `next` : 이하 생략하고 다음 반복문 실행
- `break` : 루프 탈출

예외처리

- `warning()` : warning 강제 발생
- `stop()` : error 강제 발생
- `try(expression, silent=TRUE)`: 단순 에러출력 방지
- `tryCatch(expression, warning, error, finally)` : warning/error발생시의 핸들러, final 핸들러

디버그

- `debug(function.name)` : 함수 디버그 설정
- `undebug(function.name)` : 함수 디버그 해제.
- `browser()` : 코드 내부에서 브레이크 포인트 설정
- `setBreakpoint(srcfile, line)` : 코드 외부에서 브레이크 포인트 설정

디버그 모드 명령

- `n` (next) : 다음 라인으로 진행
- `c` (continue) : 다음 브레이크포인트로 진행
- `where` : 현재 코드 표시
- `Q` (Quite) : 중지

객체지향

	S3 구현방식	S4 구현방식
클래스 정의	리스트 생성	`setClass()`
인스턴스 생성	리스트 생성	`new()`
멤버 referencing	`$`	`@`
generic 함수 나열	`methods()`	`showMethods()`
generic 함수 구현	`method.class`	`setMethod()`
generic 선언	`useMethod()`	`setGeneric()`

패키지 프로그래밍

패키지 명령

- 설치된 패키지 확인 : `library()`, `installed.packages()`
- CRAN 패키지 설치 : `install.packages("packagename")`
- Repo 패키지 설치 : `install.packages("packagename", repo=repoaddress)`
- 패키지 업데이트 : `update.packages()`
- 패키지 로드 : `library(packagename)`
- 패키지 언로드 : `detach("package:packagename", unload=TRUE)`
- 로드된 패키지 목록 : `search()`

데이터셋 명령

- 로드된 패키지 데이터셋 확인 : `data()`
- 특정 패키지 데이터셋 확인 : `data(package="packagename")`
- 데이터셋을 워크스페이스에 로드 : `data(datasetname)`

데이터 입출력

- 콘솔 입력 : `scan(what, sep)`
- text 입력 : `read.table(file, header, sep)`
- text 출력 : `write.table(x, file, sep)`

패키지 작성순서

1. package.skeleton
2. DESCRIPTION
3. R/script-name.R
4. man/document-name.Rd
5. R CMD check
6. R CMD install

시간/날짜 자료형

Date

- 문자열에서 생성 : `as.Date(date.string, format='%m/%d/%Y')`
- 문자열로 변환 : `format.Date(date, format)`
- 현재 날짜 : `Sys.Date()`
- +/- 연산 결과 : `difftime` 자료형
- `difftime` 값 : `as.numeric(dt, units)`

POSIXct/POSIXlt

- 문자열에서 생성 : `as.POSIXct(datetime, tz)`, `as.POSIXlt(datetime)`
- 문자열로 변환 : `format.POSIXct(t, format)`, `format.POSIXlt(t, format)`
- 현재 날짜시간 : `Sys.time()`

시계열 자료형

ts 클래스

- `ts(data, frequency, start, end)`
- `data` 시계열 자료
- `frequency` 연간:1, 분기:4, 월간:12
- `start` 시작 월/일자 벡터
- `start()`, `end()` : 시작일/종료일 속성
- `frequency()`, `deltat()` : 빈도 및 주기
- `window(ts, start, end)` : 일부 구간
- `lag(ts, k)` : k 단위만큼 지연
- `diff(ts, lag)` : 시계열 지연 차이
- `cbind(ts1, ts2)` : 시계열 합치기
- `ts.union(ts1, ts2)` : 시계열 합치기
- `ts.intersect(ts1, ts2)` : 데이터가 없는 부분 생략

zoo 클래스

- `zoo(data, order.by)` : 생성
- `as.zoo(data)` : 변환
- `x` : 시계열 자료
- `order.by` : 날짜/시간 인덱스
- `index()` : 인덱스 추출
- `coredata()` : 자료 추출
- `[datetime variable]` : 인덱싱

xts 클래스

- `xts(data, order.by)` : 생성
- `as.xts(data)` : 변환
- `[datetime string]` : 인덱싱

금융관련 패키지

많이 사용되는 패키지

- TTR: 기술적 분석
- quantmod: 금융 시계열 자료 임포트 및 차팅
- Quandl: 금융 시계열 자료 임포트

금융자료 임포트

- `getSymbols(symbol, src, auto.assign=TRUE)`
- src=yahoo : 야후 금융 사이트
- src=google : 구글 금융 사이트
- src=oanda : oanda 외환 사이트
- src=FRED : FRED 데이터베이스
- src=RData : RData 파일
- src=csv : csv 파일
- src=SQLite : SQLite 데이터베이스 파일
- src=MySQL : MySQL 데이터베이스